# LPCOpen Platform for LPC5411X microcontrollers

5411x

Generated by Doxygen 1.8.10

Fri Mar 4 2016 18:34:19

# Contents

# Chapter 1

# LPCOpen Platform for the NXP LPC5411X family of Microcontrollers

**Introduction**
This documentation describes the LPCOpen Platform software API and examples for NXP LPC5411X family of microcontrollers.

**LPCOpen summary page on LPCware.com**
Visit the `LPCOpen page on LPCware.com` for more resources on the LPCOpen platform.

**LPCOpen quickstart guides**
Visit the `LPCOpen quickstart guides` page to get started building LPCOpen projects.

**LPCOpen software API documentation**
Software API documentation provides API descriptions for the various drivers in LPCopen.

- Chip Driver Layer Chip specific drivers
- Board Support Layer Board support code for various support boards
- Supported RTOSes such as FreeRTOS and uC/OS-III

## Chapter 2

# MISRA-C:2004 Compliance Exceptions

CMSIS violates the following MISRA-C:2004 rules:

- Required Rule 8.5, object/function definition in header file.
  Function definitions in header files are used to allow 'inlining'.

- Required Rule 18.4, declaration of union type or object of union type: '{...}'.
  Unions are used for effective representation of core registers.

- Advisory Rule 19.7, Function-like macro defined.
  Function-like macros are used to allow more efficient code.

# Chapter 3

# Module Index

## 3.1 Modules

Here is a list of all modules:

# Chapter 4

# Data Structure Index

## 4.1 Data Structures

Here are the data structures with brief descriptions:

# Chapter 5

# File Index

## 5.1 File List

Here is a list of all files with brief descriptions:

# Chapter 6

# Module Documentation

## 6.1 BOARD: Common board components used with board drivers

## 6.2 BOARD: LPC5411X boards

### 6.2.1 Detailed Description

The following LPC5411X boards are supported in LPCOpen.

#### Modules

- NXP LPCXpresso LPC54114 LQFP board

## 6.3 Board specific drivers and support functions

### 6.3.1 Detailed Description

Board specific setup code includes code for setting up a board's default memory, clocking, muxing, or other interfaces. The default UART port for debug output is also setup as part of the board layer.

### Modules

- BOARD: Common board components used with board drivers
- BOARD: LPC5411X boards

## 6.4 CHIP: LPC5411X A/D conversion driver

### 6.4.1 Detailed Description

### Data Structures

- struct LPC_ADC_T

  *ADC register block structure.*

### Macros

- #define ADC_MAX_SAMPLE_RATE 80000000
- #define ADC_MAX_CHANNEL_NUM 12
- #define ADC_CR_CLKDIV_MASK (0xFF << 0)

  *ADC register support bitfields and mask.*
- #define ADC_CR_CLKDIV_BITPOS (0)
- #define ADC_CR_ASYNC_MODE (1 << 8)
- #define ADC_CR_RESOL(n) ((n) << 9)
- #define ADC_CR_LPWRMODEBIT (1 << 10)
- #define ADC_CR_BYPASS (1 << 11)
- #define ADC_CR_TSAMP(n) ((n) << 12)
- #define ADC_CR_CALMODEBIT (1 << 30)
- #define ADC_CR_BITACC(n) ((((n) & 0x1) << 9))
- #define ADC_CR_CLKDIV(n) ((((n) & 0xFF) << 0))
- #define ADC_SAMPLE_RATE_CONFIG_MASK (ADC_CR_CLKDIV(0xFF) | ADC_CR_BITACC(0x01))
- #define ADC_SEQ_CTRL_CHANNEL_EN(n) (1 << n)
- #define ADC_SEQ_CTRL_TRIGGER(n) ((n & 0x3f)<<12)
- #define ADC_SEQ_CTRL_HWTRIG_POLPOS (1 << 18)
- #define ADC_SEQ_CTRL_HWTRIG_SYNCBYPASS (1 << 19)
- #define ADC_SEQ_CTRL_START (1 << 26)
- #define ADC_SEQ_CTRL_BURST (1 << 27)
- #define ADC_SEQ_CTRL_SINGLESTEP (1 << 28)
- #define ADC_SEQ_CTRL_LOWPRIO (1 << 29)
- #define ADC_SEQ_CTRL_MODE_EOS (1 << 30)
- #define ADC_SEQ_CTRL_SEQ_ENA (1UL << 31)
- #define ADC_SEQ_GDAT_RESULT_MASK (0xFFF << 4)
- #define ADC_SEQ_GDAT_RESULT_BITPOS (4)
- #define ADC_SEQ_GDAT_THCMPRANGE_MASK (0x3 << 16)
- #define ADC_SEQ_GDAT_THCMPRANGE_BITPOS (16)
- #define ADC_SEQ_GDAT_THCMPCROSS_MASK (0x3 << 18)
- #define ADC_SEQ_GDAT_THCMPCROSS_BITPOS (18)
- #define ADC_SEQ_GDAT_CHAN_MASK (0xF << 26)
- #define ADC_SEQ_GDAT_CHAN_BITPOS (26)
- #define ADC_SEQ_GDAT_OVERRUN (1 << 30)
- #define ADC_SEQ_GDAT_DATAVALID (1UL << 31)
- #define ADC_DR_RESULT_BITPOS (4)
- #define ADC_DR_RESULT(n) ((((n) >> 4) & 0xFFF))
- #define ADC_DR_THCMPRANGE_MASK (0x3 << 16)
- #define ADC_DR_THCMPRANGE_BITPOS (16)
- #define ADC_DR_THCMPRANGE(n) (((n) >> ADC_DR_THCMPRANGE_BITPOS) & 0x3)
- #define ADC_DR_THCMPCROSS_MASK (0x3 << 18)
- #define ADC_DR_THCMPCROSS_BITPOS (18)
- #define ADC_DR_THCMPCROSS(n) (((n) >> ADC_DR_THCMPCROSS_BITPOS) & 0x3)

- #define ADC_DR_CHAN_MASK (0xF << 26)
- #define ADC_DR_CHAN_BITPOS (26)
- #define ADC_DR_CHANNEL(n) (((n) >> ADC_DR_CHAN_BITPOS) & 0xF)
- #define ADC_DR_OVERRUN (1 << 30)
- #define ADC_DR_DATAVALID (1UL << 31)
- #define ADC_DR_DONE(n) (((n) >> 31))
- #define ADC_THR_VAL_MASK (0xFFF << 4)
- #define ADC_THR_VAL_POS (4)
- #define ADC_THRSEL_CHAN_SEL_THR1(n) (1 << (n))
- #define ADC_INTEN_SEQA_ENABLE (1 << 0)
- #define ADC_INTEN_SEQB_ENABLE (1 << 1)
- #define ADC_INTEN_SEQN_ENABLE(seq) (1 << (seq))
- #define ADC_INTEN_OVRRUN_ENABLE (1 << 2)
- #define ADC_INTEN_CMP_DISBALE (0)
- #define ADC_INTEN_CMP_OUTSIDETH (1)
- #define ADC_INTEN_CMP_CROSSTH (2)
- #define ADC_INTEN_CMP_MASK (3)
- #define ADC_INTEN_CMP_ENABLE(isel, ch) (((isel) & ADC_INTEN_CMP_MASK) << ((2 ∗ (ch)) + 3))
- #define ADC_FLAGS_THCMP_MASK(ch) (1 << (ch))
- #define ADC_FLAGS_OVRRUN_MASK(ch) (1 << (12 + (ch)))
- #define ADC_FLAGS_SEQA_OVRRUN_MASK (1 << 24)
- #define ADC_FLAGS_SEQB_OVRRUN_MASK (1 << 25)
- #define ADC_FLAGS_SEQN_OVRRUN_MASK(seq) (1 << (24 + (seq)))
- #define ADC_FLAGS_SEQA_INT_MASK (1 << 28)
- #define ADC_FLAGS_SEQB_INT_MASK (1 << 29)
- #define ADC_FLAGS_SEQN_INT_MASK(seq) (1 << (28 + (seq)))
- #define ADC_FLAGS_THCMP_INT_MASK (1 << 30)
- #define ADC_FLAGS_OVRRUN_INT_MASK (1UL << 31)
- #define ADC_STARTUP_ENABLE (0x1 << 0)
- #define ADC_STARTUP_INIT (0x1 << 1)
- #define ADC_CALIB (0x1<<0)
- #define ADC_CALREQD (0x1<<1)

## Enumerations

- enum ADC_SEQ_IDX_T { ADC_SEQA_IDX = 0, ADC_SEQB_IDX }
- enum ADC_TSAMP_T {
  ADC_TSAMP_2CLK5 = 0, ADC_TSAMP_3CLK5, ADC_TSAMP_4CLK5, ADC_TSAMP_5CLK5,
  ADC_TSAMP_6CLK5, ADC_TSAMP_7CLK5, ADC_TSAMP_8CLK5, ADC_TSAMP_9CLK5 }

  *ADC sampling time bits 12, 13 and 14.*
- enum ADC_DR_THCMPRANGE_T { ADC_DR_THCMPRANGE_INRANGE, ADC_DR_THCMPRANGE_RESERVED, ADC_DR_THCMPRANGE_BELOW, ADC_DR_THCMPRANGE_ABOVE }
- enum ADC_DR_THCMPCROSS_T { ADC_DR_THCMPCROSS_NOCROSS, ADC_DR_THCMPCROSS_RESERVED, ADC_DR_THCMPCROSS_DOWNWARD, ADC_DR_THCMPCROSS_UPWARD }
- enum ADC_INTEN_THCMP_T { ADC_INTEN_THCMP_DISABLE, ADC_INTEN_THCMP_OUTSIDE, ADC_INTEN_THCMP_CROSSING }

## Functions

- void Chip_ADC_Init (LPC_ADC_T ∗pADC, uint32_t flags)

  *Initialize the ADC peripheral.*
- void Chip_ADC_DeInit (LPC_ADC_T ∗pADC)

  *Shutdown ADC.*
- __STATIC_INLINE void Chip_ADC_SetDivider (LPC_ADC_T ∗pADC, uint8_t div)

  *Set ADC divider.*
- void Chip_ADC_SetClockRate (LPC_ADC_T ∗pADC, uint32_t rate)

  *Set ADC clock rate.*
- __STATIC_INLINE uint8_t Chip_ADC_GetDivider (LPC_ADC_T ∗pADC)

  *Get ADC divider.*
- uint32_t Chip_ADC_Calibration (LPC_ADC_T ∗pADC)

  *Perform ADC calibration.*
- __STATIC_INLINE void Chip_ADC_SelectTempSensorInput (LPC_ADC_T ∗pADC)

  *Selects Temperature sensor as the input for Channel 0.*
- __STATIC_INLINE void Chip_ADC_SetSequencerBits (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex, uint32_t bits)

  *Helper function for safely setting ADC sequencer register bits.*
- __STATIC_INLINE void Chip_ADC_ClearSequencerBits (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex, uint32_t bits)

  *Helper function for safely clearing ADC sequencer register bits.*
- __STATIC_INLINE void Chip_ADC_SetupSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex, uint32_t options)

  *Sets up ADC conversion sequencer A or B.*
- __STATIC_INLINE uint32_t Chip_ADC_GetSequencerCtrl (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seq←Index)

  *Get sequenceX control register value.*
- __STATIC_INLINE void Chip_ADC_EnableSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

  *Enables a sequencer.*
- __STATIC_INLINE void Chip_ADC_DisableSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

  *Disables a sequencer.*
- __STATIC_INLINE void Chip_ADC_StartSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

  *Forces a sequencer trigger event (software trigger of ADC)*
- __STATIC_INLINE void Chip_ADC_StartBurstSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seq←Index)

  *Starts sequencer burst mode.*
- __STATIC_INLINE void Chip_ADC_StopBurstSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seq←Index)

  *Stops sequencer burst mode.*
- __STATIC_INLINE uint32_t Chip_ADC_GetGlobalDataReg (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seq←Index)

  *Read a ADC sequence global data register.*
- __STATIC_INLINE uint32_t Chip_ADC_GetDataReg (LPC_ADC_T ∗pADC, uint8_t index)

  *Read a ADC data register.*
- __STATIC_INLINE void Chip_ADC_SetThrLowValue (LPC_ADC_T ∗pADC, uint8_t thrnum, uint16_t value)

  *Set Threshold low value in ADC.*
- __STATIC_INLINE void Chip_ADC_SetThrHighValue (LPC_ADC_T ∗pADC, uint8_t thrnum, uint16_t value)

  *Set Threshold high value in ADC.*
- __STATIC_INLINE void Chip_ADC_SelectTH0Channels (LPC_ADC_T ∗pADC, uint32_t channels)

  *Select threshold 0 values for comparison for selected channels.*
- __STATIC_INLINE void Chip_ADC_SelectTH1Channels (LPC_ADC_T ∗pADC, uint32_t channels)

*Select threshold 1 value for comparison for selected channels.*

- __STATIC_INLINE void Chip_ADC_EnableInt (LPC_ADC_T ∗pADC, uint32_t intMask)

  *Enable interrupts in ADC (sequencers A/B and overrun)*
- __STATIC_INLINE void Chip_ADC_DisableInt (LPC_ADC_T ∗pADC, uint32_t intMask)

  *Disable interrupts in ADC (sequencers A/B and overrun)*
- __STATIC_INLINE void Chip_ADC_SetThresholdInt (LPC_ADC_T ∗pADC, uint8_t ch, ADC_INTEN_THC↩ MP_T thInt)

  *Enable a threshold event interrupt in ADC.*
- __STATIC_INLINE uint32_t Chip_ADC_GetFlags (LPC_ADC_T ∗pADC)

  *Get flags register in ADC.*
- __STATIC_INLINE void Chip_ADC_ClearFlags (LPC_ADC_T ∗pADC, uint32_t flags)

  *Clear flags register in ADC.*
- __STATIC_INLINE void Chip_ADC_SetTHRSELBits (LPC_ADC_T ∗pADC, uint32_t mask)

  *Set Threshold selection bits.*
- __STATIC_INLINE void Chip_ADC_ClearTHRSELBits (LPC_ADC_T ∗pADC, uint32_t mask)

  *Clear Threshold selection bits.*

## 6.4.2 Macro Definition Documentation

### 6.4.2.1 #define ADC_CALIB (0x1<<0)

Definition at line 195 of file adc_5411x.h.

### 6.4.2.2 #define ADC_CALREQD (0x1<<1)

Definition at line 196 of file adc_5411x.h.

### 6.4.2.3 #define ADC_CR_ASYNC_MODE (1 << 8)

Asynchronous mode enable bit

Definition at line 96 of file adc_5411x.h.

### 6.4.2.4 #define ADC_CR_BITACC( *n* ) ((((n) & 0x1) << 9))

12-bit or 10-bit ADC accuracy

Definition at line 102 of file adc_5411x.h.

### 6.4.2.5 #define ADC_CR_BYPASS (1 << 11)

Bypass mode

Definition at line 99 of file adc_5411x.h.

### 6.4.2.6 #define ADC_CR_CALMODEBIT (1 << 30)

Self calibration cycle enable bit

Definition at line 101 of file adc_5411x.h.

#### 6.4.2.7 #define ADC_CR_CLKDIV( n ) ((((n) & 0xFF) << 0))

The APB clock (PCLK) is divided by (this value plus one) to produce the clock for the A/D

Definition at line 103 of file adc_5411x.h.

#### 6.4.2.8 #define ADC_CR_CLKDIV_BITPOS (0)

Bit position for Clock divider value

Definition at line 95 of file adc_5411x.h.

#### 6.4.2.9 #define ADC_CR_CLKDIV_MASK (0xFF << 0)

ADC register support bitfields and mask.

ADC Control register bit fields Mask for Clock divider value

Definition at line 94 of file adc_5411x.h.

#### 6.4.2.10 #define ADC_CR_LPWRMODEBIT (1 << 10)

Low power mode enable bit

Definition at line 98 of file adc_5411x.h.

#### 6.4.2.11 #define ADC_CR_RESOL( n ) ((n) << 9)

2-bits, 6(0x0),8(0x1),10(0x2),12(0x3)-bit mode enable bit

Definition at line 97 of file adc_5411x.h.

#### 6.4.2.12 #define ADC_CR_TSAMP( n ) ((n) << 12)

3-bits, 2.5(0x0),3.5(0x1),4.5(0x2),5.5(0x3),6.5(0x4),7.5(0x5),8.5(0x6),9.5(0x7) ADC clocks sampling time

Definition at line 100 of file adc_5411x.h.

#### 6.4.2.13 #define ADC_DR_CHAN_BITPOS (26)

Channel number bit position

Definition at line 154 of file adc_5411x.h.

#### 6.4.2.14 #define ADC_DR_CHAN_MASK (0xF << 26)

Channel number mask

Definition at line 153 of file adc_5411x.h.

#### 6.4.2.15 #define ADC_DR_CHANNEL( n ) (((n) >> ADC_DR_CHAN_BITPOS) & 0xF)

Channel number bit position

Definition at line 155 of file adc_5411x.h.

#### 6.4.2.16 #define ADC_DR_DATAVALID (1UL << 31)

Data valid bit

Definition at line 157 of file adc_5411x.h.

#### 6.4.2.17 #define ADC_DR_DONE( *n* ) (((n) >> 31))

Definition at line 158 of file adc_5411x.h.

#### 6.4.2.18 #define ADC_DR_OVERRUN (1 << 30)

Overrun bit

Definition at line 156 of file adc_5411x.h.

#### 6.4.2.19 #define ADC_DR_RESULT( *n* ) ((((n) >> 4) & 0xFFF))

Macro for getting the ADC data value

Definition at line 146 of file adc_5411x.h.

#### 6.4.2.20 #define ADC_DR_RESULT_BITPOS (4)

ADC Data register bit fields Result start bit position

Definition at line 145 of file adc_5411x.h.

#### 6.4.2.21 #define ADC_DR_THCMPCROSS( *n* ) (((n) >> ADC_DR_THCMPCROSS_BITPOS) & 0x3)

Definition at line 152 of file adc_5411x.h.

#### 6.4.2.22 #define ADC_DR_THCMPCROSS_BITPOS (18)

Comparison cross bit position

Definition at line 151 of file adc_5411x.h.

#### 6.4.2.23 #define ADC_DR_THCMPCROSS_MASK (0x3 << 18)

Comparion cross mask

Definition at line 150 of file adc_5411x.h.

#### 6.4.2.24 #define ADC_DR_THCMPRANGE( *n* ) (((n) >> ADC_DR_THCMPRANGE_BITPOS) & 0x3)

Definition at line 149 of file adc_5411x.h.

#### 6.4.2.25 #define ADC_DR_THCMPRANGE_BITPOS (16)

Comparison range bit position

Definition at line 148 of file adc_5411x.h.

#### 6.4.2.26 #define ADC_DR_THCMPRANGE_MASK (0x3 << 16)

Comparion range mask

Definition at line 147 of file adc_5411x.h.

#### 6.4.2.27 #define ADC_FLAGS_OVRRUN_INT_MASK (1UL << 31)

Overrun Interrupt status

Definition at line 188 of file adc_5411x.h.

#### 6.4.2.28 #define ADC_FLAGS_OVRRUN_MASK( *ch* ) (1 << (12 + (ch)))

Overrun status for channel

Definition at line 180 of file adc_5411x.h.

#### 6.4.2.29 #define ADC_FLAGS_SEQA_INT_MASK (1 << 28)

Seq A Interrupt status

Definition at line 184 of file adc_5411x.h.

#### 6.4.2.30 #define ADC_FLAGS_SEQA_OVRRUN_MASK (1 << 24)

Seq A Overrun status

Definition at line 181 of file adc_5411x.h.

#### 6.4.2.31 #define ADC_FLAGS_SEQB_INT_MASK (1 << 29)

Seq B Interrupt status

Definition at line 185 of file adc_5411x.h.

#### 6.4.2.32 #define ADC_FLAGS_SEQB_OVRRUN_MASK (1 << 25)

Seq B Overrun status

Definition at line 182 of file adc_5411x.h.

#### 6.4.2.33 #define ADC_FLAGS_SEQN_INT_MASK( *seq* ) (1 << (28 + (seq)))

Seq A/B Interrupt status

Definition at line 186 of file adc_5411x.h.

#### 6.4.2.34 #define ADC_FLAGS_SEQN_OVRRUN_MASK( *seq* ) (1 << (24 + (seq)))

Seq A/B Overrun status

Definition at line 183 of file adc_5411x.h.

#### 6.4.2.35 #define ADC_FLAGS_THCMP_INT_MASK (1 $<<$ 30)

Threshold comparison Interrupt status

Definition at line 187 of file adc_5411x.h.

#### 6.4.2.36 #define ADC_FLAGS_THCMP_MASK( *ch* ) (1 $<<$ (ch))

ADC Flags register bit fields Threshold comparison status for channel

Definition at line 179 of file adc_5411x.h.

#### 6.4.2.37 #define ADC_INTEN_CMP_CROSSTH (2)

Crossing threshold interrupt value

Definition at line 174 of file adc_5411x.h.

#### 6.4.2.38 #define ADC_INTEN_CMP_DISBALE (0)

Disable comparison interrupt value

Definition at line 172 of file adc_5411x.h.

#### 6.4.2.39 #define ADC_INTEN_CMP_ENABLE( *isel, ch* ) (((isel) & ADC_INTEN_CMP_MASK) $<<$ ((2 $*$ (ch)) + 3))

Interrupt selection for channel

Definition at line 176 of file adc_5411x.h.

#### 6.4.2.40 #define ADC_INTEN_CMP_MASK (3)

Comparison interrupt value mask

Definition at line 175 of file adc_5411x.h.

#### 6.4.2.41 #define ADC_INTEN_CMP_OUTSIDETH (1)

Outside threshold interrupt value

Definition at line 173 of file adc_5411x.h.

#### 6.4.2.42 #define ADC_INTEN_OVRRUN_ENABLE (1 $<<$ 2)

Overrun Interrupt enable bit

Definition at line 171 of file adc_5411x.h.

#### 6.4.2.43 #define ADC_INTEN_SEQA_ENABLE (1 $<<$ 0)

ADC Interrupt Enable register bit fields Sequence A Interrupt enable bit

Definition at line 168 of file adc_5411x.h.

#### 6.4.2.44 #define ADC_INTEN_SEQB_ENABLE (1 << 1)

Sequence B Interrupt enable bit

Definition at line 169 of file adc_5411x.h.

#### 6.4.2.45 #define ADC_INTEN_SEQN_ENABLE( *seq* ) (1 << (seq))

Sequence A/B Interrupt enable bit

Definition at line 170 of file adc_5411x.h.

#### 6.4.2.46 #define ADC_MAX_CHANNEL_NUM 12

Definition at line 88 of file adc_5411x.h.

#### 6.4.2.47 #define ADC_MAX_SAMPLE_RATE 80000000

Maximum sample rate in Hz (12-bit conversions)

Definition at line 87 of file adc_5411x.h.

#### 6.4.2.48 #define ADC_SAMPLE_RATE_CONFIG_MASK (ADC_CR_CLKDIV(0xFF) | ADC_CR_BITACC(0x01))

Definition at line 104 of file adc_5411x.h.

#### 6.4.2.49 #define ADC_SEQ_CTRL_BURST (1 << 27)

Repeated conversion enable bit

Definition at line 126 of file adc_5411x.h.

#### 6.4.2.50 #define ADC_SEQ_CTRL_CHANNEL_EN( *n* ) (1 << n)

SEQ_CTRL register bit fields

Definition at line 121 of file adc_5411x.h.

#### 6.4.2.51 #define ADC_SEQ_CTRL_HWTRIG_POLPOS (1 << 18)

HW trigger polarity - positive edge

Definition at line 123 of file adc_5411x.h.

#### 6.4.2.52 #define ADC_SEQ_CTRL_HWTRIG_SYNCBYPASS (1 << 19)

HW trigger bypass synchronisation

Definition at line 124 of file adc_5411x.h.

#### 6.4.2.53 #define ADC_SEQ_CTRL_LOWPRIO (1 << 29)

High priority enable bit (regardless of name)

Definition at line 128 of file adc_5411x.h.

#### 6.4.2.54 #define ADC_SEQ_CTRL_MODE_EOS (1 << 30)

Mode End of sequence enable bit

Definition at line 129 of file adc_5411x.h.

#### 6.4.2.55 #define ADC_SEQ_CTRL_SEQ_ENA (1UL << 31)

Sequence enable bit

Definition at line 130 of file adc_5411x.h.

#### 6.4.2.56 #define ADC_SEQ_CTRL_SINGLESTEP (1 << 28)

Single step enable bit

Definition at line 127 of file adc_5411x.h.

#### 6.4.2.57 #define ADC_SEQ_CTRL_START (1 << 26)

Start conversion enable bit

Definition at line 125 of file adc_5411x.h.

#### 6.4.2.58 #define ADC_SEQ_CTRL_TRIGGER( *n* ) ((n & 0x3f)<<12)

Definition at line 122 of file adc_5411x.h.

#### 6.4.2.59 #define ADC_SEQ_GDAT_CHAN_BITPOS (26)

Channel number bit position

Definition at line 140 of file adc_5411x.h.

#### 6.4.2.60 #define ADC_SEQ_GDAT_CHAN_MASK (0xF << 26)

Channel number mask

Definition at line 139 of file adc_5411x.h.

#### 6.4.2.61 #define ADC_SEQ_GDAT_DATAVALID (1UL << 31)

Data valid bit

Definition at line 142 of file adc_5411x.h.

#### 6.4.2.62 #define ADC_SEQ_GDAT_OVERRUN (1 << 30)

Overrun bit

Definition at line 141 of file adc_5411x.h.

#### 6.4.2.63 #define ADC_SEQ_GDAT_RESULT_BITPOS (4)

Result start bit position

Definition at line 134 of file adc_5411x.h.

#### 6.4.2.64 #define ADC_SEQ_GDAT_RESULT_MASK (0xFFF $<<$ 4)

ADC global data register bit fields Result value mask

Definition at line 133 of file adc_5411x.h.

#### 6.4.2.65 #define ADC_SEQ_GDAT_THCMPCROSS_BITPOS (18)

Comparison cross bit position

Definition at line 138 of file adc_5411x.h.

#### 6.4.2.66 #define ADC_SEQ_GDAT_THCMPCROSS_MASK (0x3 $<<$ 18)

Comparion cross mask

Definition at line 137 of file adc_5411x.h.

#### 6.4.2.67 #define ADC_SEQ_GDAT_THCMPRANGE_BITPOS (16)

Comparison range bit position

Definition at line 136 of file adc_5411x.h.

#### 6.4.2.68 #define ADC_SEQ_GDAT_THCMPRANGE_MASK (0x3 $<<$ 16)

Comparion range mask

Definition at line 135 of file adc_5411x.h.

#### 6.4.2.69 #define ADC_STARTUP_ENABLE (0x1 $<<$ 0)

ADC Startup register bit fields

Definition at line 191 of file adc_5411x.h.

#### 6.4.2.70 #define ADC_STARTUP_INIT (0x1 $<<$ 1)

Definition at line 192 of file adc_5411x.h.

#### 6.4.2.71 #define ADC_THR_VAL_MASK (0xFFF $<<$ 4)

ADC low/high Threshold register bit fields Threshold value bit mask

Definition at line 161 of file adc_5411x.h.

#### 6.4.2.72 #define ADC_THR_VAL_POS (4)

Threshold value bit position

Definition at line 162 of file adc_5411x.h.

#### 6.4.2.73 #define ADC_THRSEL_CHAN_SEL_THR1( *n* ) (1 << (n))

ADC Threshold select register bit fields Select THR1 register for channel n

Definition at line 165 of file adc_5411x.h.

### 6.4.3 Enumeration Type Documentation

#### 6.4.3.1 enum ADC_DR_THCMPCROSS_T

ADC sequence global data register threshold comparison cross enumerations

**Enumerator**

***ADC_DR_THCMPCROSS_NOCROSS***

***ADC_DR_THCMPCROSS_RESERVED***

***ADC_DR_THCMPCROSS_DOWNWARD***

***ADC_DR_THCMPCROSS_UPWARD***

Definition at line 425 of file adc_5411x.h.

#### 6.4.3.2 enum ADC_DR_THCMPRANGE_T

ADC sequence global data register threshold comparison range enumerations

**Enumerator**

***ADC_DR_THCMPRANGE_INRANGE***

***ADC_DR_THCMPRANGE_RESERVED***

***ADC_DR_THCMPRANGE_BELOW***

***ADC_DR_THCMPRANGE_ABOVE***

Definition at line 417 of file adc_5411x.h.

#### 6.4.3.3 enum ADC_INTEN_THCMP_T

Threshold interrupt event options

**Enumerator**

***ADC_INTEN_THCMP_DISABLE***

***ADC_INTEN_THCMP_OUTSIDE***

***ADC_INTEN_THCMP_CROSSING***

Definition at line 559 of file adc_5411x.h.

#### 6.4.3.4 enum ADC_SEQ_IDX_T

Sequence index enumerations, used in various parts of the code for register indexing and sequencer selection

**Enumerator**

***ADC_SEQA_IDX***

***ADC_SEQB_IDX***

Definition at line 46 of file adc_5411x.h.

#### 6.4.3.5 enum ADC_TSAMP_T

ADC sampling time bits 12, 13 and 14.

**Enumerator**

***ADC_TSAMP_2CLK5***

***ADC_TSAMP_3CLK5***

***ADC_TSAMP_4CLK5***

***ADC_TSAMP_5CLK5***

***ADC_TSAMP_6CLK5***

***ADC_TSAMP_7CLK5***

***ADC_TSAMP_8CLK5***

***ADC_TSAMP_9CLK5***

Definition at line 109 of file adc_5411x.h.

### 6.4.4 Function Documentation

#### 6.4.4.1 uint32_t Chip_ADC_Calibration ( LPC_ADC_T ∗ *pADC* )

Perform ADC calibration.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |

**Returns**

LPC_OK on success, ERR_TIME_OUT or ERR_ADC_NO_POWER on failure

**Note**

Calibration is not done as part of Chip_ADC_Init(), but is required after the call to Chip_ADC_Init() or after returning from a power-down state.

Definition at line 76 of file adc_5411x.c.

#### 6.4.4.2 __STATIC_INLINE void Chip_ADC_ClearFlags ( LPC_ADC_T ∗ *pADC,* uint32_t *flags* )

Clear flags register in ADC.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *flags* | : An Or'ed values of ADC_FLAGS_∗ values to clear |

**Returns**

Flags register value (ORed ADC_FLAG∗ values)

Definition at line 603 of file adc_5411x.h.

#### 6.4.4.3 __STATIC_INLINE void Chip_ADC_ClearSequencerBits ( LPC_ADC_T ∗ *pADC*, ADC_SEQ_IDX_T *seqIndex*, uint32_t *bits* )

Helper function for safely clearing ADC sequencer register bits.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to clear bits for |
| *bits* | : Or'ed bits of a sequencer register to clear |

**Returns**

Nothing

**Note**

This function will safely clear the ADC sequencer register bits while maintaining bits 20..25 as 0, regardless of the read state of those bits.

Definition at line 306 of file adc_5411x.h.

#### 6.4.4.4 __STATIC_INLINE void Chip_ADC_ClearTHRSELBits ( LPC_ADC_T ∗ *pADC*, uint32_t *mask* )

Clear Threshold selection bits.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *mask* | : Threshold selection mask |

**Returns**

Nothing

Definition at line 624 of file adc_5411x.h.

#### 6.4.4.5 void Chip_ADC_DeInit ( LPC_ADC_T ∗ *pADC* )

Shutdown ADC.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |

**Returns**

Nothing

**Note**

Disables the ADC clocks and ADC power

Definition at line 65 of file adc_5411x.c.

#### 6.4.4.6 __STATIC_INLINE void Chip_ADC_DisableInt ( LPC_ADC_T ∗ *pADC,* uint32_t *intMask* )

Disable interrupts in ADC (sequencers A/B and overrun)

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *intMask* | : Interrupt values to be disabled (see notes) |

**Returns**

None

**Note**

Select one or more OR'ed values of ADC_INTEN_SEQA_ENABLE, ADC_INTEN_SEQB_ENABLE, and A←DC_INTEN_OVRRUN_ENABLE to disable the specific ADC interrupts.

Definition at line 552 of file adc_5411x.h.

#### 6.4.4.7 __STATIC_INLINE void Chip_ADC_DisableSequencer ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex* )

Disables a sequencer.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to disable |

**Returns**

Nothing

Definition at line 368 of file adc_5411x.h.

#### 6.4.4.8 __STATIC_INLINE void Chip_ADC_EnableInt ( LPC_ADC_T ∗ *pADC,* uint32_t *intMask* )

Enable interrupts in ADC (sequencers A/B and overrun)

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *intMask* | : Interrupt values to be enabled (see notes) |

**Returns**

None

**Note**

Select one or more OR'ed values of ADC_INTEN_SEQA_ENABLE, ADC_INTEN_SEQB_ENABLE, and ADC_INTEN_OVRRUN_ENABLE to enable the specific ADC interrupts.

Definition at line 537 of file adc_5411x.h.

### 6.4.4.9 __STATIC_INLINE void Chip_ADC_EnableSequencer ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex* )

Enables a sequencer.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to enable |

**Returns**

Nothing

Definition at line 357 of file adc_5411x.h.

### 6.4.4.10 __STATIC_INLINE uint32_t Chip_ADC_GetDataReg ( LPC_ADC_T ∗ *pADC,* uint8_t *index* )

Read a ADC data register.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *index* | : Data register to read, 0-11 |

**Returns**

Current raw value of the ADC data register

**Note**

This function returns the raw value of the data register and clears the overrun and datavalid status for the register. Once this register is read, the following functions can be used to parse the raw value:
uint32_t adcDataRawValue = Chip_ADC_GetDataReg(LPC_ADC, ADC_MAX_CHANNEL_NUM); // Get raw value uint32_t adcDataValue = ADC_DR_RESULT(adcDataRawValue); // Aligned and masked ADC data value ADC_DR_THCMPRANGE_T adcRange = (ADC_DR_THCMPRANGE_T) ADC_DR_THCMPRANGE(adcDataRawValue); // Sample range compared to threshold low/high ADC_DR_THCMPCROSS_T adcRange = (ADC_DR_THCMPCROSS_T) ADC_DR_THCMPCROSS(adcDataRawValue); // Sample cross compared to threshold low uint32_t channel = ADC_DR_CHANNEL(adcDataRawValue); // ADC channel for this sample/data bool adcDataOverrun = (bool) ((adcDataRawValue & ADC_DR_OVERRUN) != 0); // Data overrun flag bool adcDataValid = (bool) ((adcDataRawValue & ADC_SEQ_GDAT_DATAVALID) != 0); // Data valid flag

Definition at line 469 of file adc_5411x.h.

#### 6.4.4.11 __STATIC_INLINE uint8_t Chip_ADC_GetDivider ( LPC_ADC_T ∗ *pADC* )

Get ADC divider.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |

**Returns**

the current ADC divider

**Note**

This function returns the divider that is used to generate the ADC frequency. The returned value must be incremented by 1. The frequency can be determined with the following function:
adc_freq = Chip_Clock_GetSystemClockRate() / (Chip_ADC_GetDivider(LPC_ADC) + 1);

Definition at line 255 of file adc_5411x.h.

#### 6.4.4.12 __STATIC_INLINE uint32_t Chip_ADC_GetFlags ( LPC_ADC_T ∗ *pADC* )

Get flags register in ADC.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |

**Returns**

Flags register value (ORed ADC_FLAG∗ values)

**Note**

Mask the return value of this function with the ADC_FLAGS_∗ definitions to determine the overall ADC interrupt events.
Example:
if (Chip_ADC_GetFlags(LPC_ADC) & ADC_FLAGS_THCMP_MASK(3) // Check of threshold comp status for ADC channel 3

Definition at line 592 of file adc_5411x.h.

#### 6.4.4.13 __STATIC_INLINE uint32_t Chip_ADC_GetGlobalDataReg ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex* )

Read a ADC sequence global data register.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to read |

**Returns**

Current raw value of the ADC sequence A or B global data register

**Note**

> This function returns the raw value of the data register and clears the overrun and datavalid status for the register. Once this register is read, the following functions can be used to parse the raw value:
> uint32_t adcDataRawValue = Chip_ADC_GetGlobalDataReg(LPC_ADC, ADC_SEQA_IDX); // Get raw value uint32_t adcDataValue = ADC_DR_RESULT(adcDataRawValue); // Aligned and masked ADC data value ADC_DR_THCMPRANGE_T adcRange = (ADC_DR_THCMPRANGE_T) ADC_DR_THCMPRANGE(adc↩DataRawValue); // Sample range compared to threshold low/high ADC_DR_THCMPCROSS_T adcRange = (ADC_DR_THCMPCROSS_T) ADC_DR_THCMPCROSS(adcDataRawValue); // Sample cross compared to threshold low uint32_t channel = ADC_DR_CHANNEL(adcDataRawValue); // ADC channel for this sample/data bool adcDataOverrun = (bool) ((adcDataRawValue & ADC_DR_OVERRUN) != 0); // Data overrun flag bool adcDataValid = (bool) ((adcDataRawValue & ADC_SEQ_GDAT_DATAVALID) != 0); // Data valid flag

Definition at line 448 of file adc_5411x.h.

#### 6.4.4.14 __STATIC_INLINE uint32_t Chip_ADC_GetSequencerCtrl ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex* )

Get sequenceX control register value.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to setup |

**Returns**

> Sequencer control register value

Definition at line 346 of file adc_5411x.h.

#### 6.4.4.15 void Chip_ADC_Init ( LPC_ADC_T ∗ *pADC,* uint32_t *flags* )

Initialize the ADC peripheral.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *flags* | : ADC flags for init (ADC_CR_MODE10BIT and/or ADC_CR_LPWRMODEBIT) |

**Returns**

> Nothing

**Note**

> To select low-power ADC mode, enable the ADC_CR_LPWRMODEBIT flag. To select 10-bit conversion mode, enable the ADC_CR_MODE10BIT flag.
> Example: Chip_ADC_Init(LPC_ADC, (ADC_CR_MODE10BIT | ADC_CR_LPWRMODEBIT));

Definition at line 51 of file adc_5411x.c.

#### 6.4.4.16 __STATIC_INLINE void Chip_ADC_SelectTempSensorInput ( LPC_ADC_T ∗ *pADC* )

Selects Temperature sensor as the input for Channel 0.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |

**Returns**

Nothing

Definition at line 275 of file adc_5411x.h.

#### 6.4.4.17 __STATIC_INLINE void Chip_ADC_SelectTH0Channels ( LPC_ADC_T ∗ *pADC*, uint32_t *channels* )

Select threshold 0 values for comparison for selected channels.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *channels* | : An OR'ed value of one or more ADC_THRSEL_CHAN_SEL_THR1(ch) values |

**Returns**

None

**Note**

Select multiple channels to use the threshold 0 comparison.
Example:
Chip_ADC_SelectTH0Channels(LPC_ADC, (ADC_THRSEL_CHAN_SEL_THR1(1) | ADC_THRSEL_CHAN_SEL_THR1(2))); // Selects channels 1 and 2 for threshold 0

Definition at line 507 of file adc_5411x.h.

#### 6.4.4.18 __STATIC_INLINE void Chip_ADC_SelectTH1Channels ( LPC_ADC_T ∗ *pADC*, uint32_t *channels* )

Select threshold 1 value for comparison for selected channels.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *channels* | : An OR'ed value of one or more ADC_THRSEL_CHAN_SEL_THR1(ch) values |

**Returns**

None

**Note**

Select multiple channels to use the 1 threshold comparison.
Example:
Chip_ADC_SelectTH1Channels(LPC_ADC, (ADC_THRSEL_CHAN_SEL_THR1(4) | ADC_THRSEL_CHAN_SEL_THR1(5))); // Selects channels 4 and 5 for 1 threshold

Definition at line 522 of file adc_5411x.h.

#### 6.4.4.19 void Chip_ADC_SetClockRate ( LPC_ADC_T ∗ *pADC*, uint32_t *rate* )

Set ADC clock rate.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *rate* | : rate in Hz to set ADC clock to (maximum ADC_MAX_SAMPLE_RATE) |

**Returns**

Nothing

Definition at line 116 of file adc_5411x.c.

### 6.4.4.20 __STATIC_INLINE void Chip_ADC_SetDivider ( LPC_ADC_T ∗ *pADC,* uint8_t *div* )

Set ADC divider.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *div* | : ADC divider value to set minus 1 |

**Returns**

Nothing

**Note**

The value is used as a divider to generate the ADC clock rate from the ADC input clock. The ADC input clock is based on the system clock. Valid values for this function are from 0 to 255 with 0=divide by 1, 1=divide by 2, 2=divide by 3, etc.
Do not decrement this value by 1.
To set the ADC clock rate to 1MHz, use the following function:
Chip_ADC_SetDivider(LPC_ADC, (Chip_Clock_GetSystemClockRate() / 1000000) - 1);

Definition at line 230 of file adc_5411x.h.

### 6.4.4.21 __STATIC_INLINE void Chip_ADC_SetSequencerBits ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex,* uint32_t *bits* )

Helper function for safely setting ADC sequencer register bits.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to set bits for |
| *bits* | : Or'ed bits of a sequencer register to set |

**Returns**

Nothing

**Note**

This function will safely set the ADC sequencer register bits while maintaining bits 20..25 as 0, regardless of the read state of those bits.

Definition at line 291 of file adc_5411x.h.

#### 6.4.4.22 __STATIC_INLINE void Chip_ADC_SetThresholdInt ( LPC_ADC_T ∗ *pADC*, uint8_t *ch*, ADC_INTEN_THCMP_T *thInt* )

Enable a threshold event interrupt in ADC.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *ch* | : ADC channel to set threshold inetrrupt for, 1-8 |
| *thInt* | : Selected threshold interrupt type |

**Returns**

None

Definition at line 572 of file adc_5411x.h.

#### 6.4.4.23 __STATIC_INLINE void Chip_ADC_SetThrHighValue ( LPC_ADC_T ∗ *pADC,* uint8_t *thrnum,* uint16_t *value* )

Set Threshold high value in ADC.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *thrnum* | : Threshold register value (1 for threshold register 1, 0 for threshold register 0) |
| *value* | : Threshold high data value (should be 12-bit value) |

**Returns**

None

Definition at line 493 of file adc_5411x.h.

#### 6.4.4.24 __STATIC_INLINE void Chip_ADC_SetThrLowValue ( LPC_ADC_T ∗ *pADC,* uint8_t *thrnum,* uint16_t *value* )

Set Threshold low value in ADC.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *thrnum* | : Threshold register value (1 for threshold register 1, 0 for threshold register 0) |
| *value* | : Threshold low data value (should be 12-bit value) |

**Returns**

None

Definition at line 481 of file adc_5411x.h.

#### 6.4.4.25 __STATIC_INLINE void Chip_ADC_SetTHRSELBits ( LPC_ADC_T ∗ *pADC,* uint32_t *mask* )

Set Threshold selection bits.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *mask* | : Threshold selection mask |

**Returns**

Nothing

Definition at line 614 of file adc_5411x.h.

#### 6.4.4.26 __STATIC_INLINE void Chip_ADC_SetupSequencer ( LPC_ADC_T ∗ *pADC*, ADC_SEQ_IDX_T *seqIndex*, uint32_t *options* )

Sets up ADC conversion sequencer A or B.

**Parameters**

| | |
|---:|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to setup |
| *options* | : OR'ed Sequencer options to setup (see notes) |

**Returns**

Nothing

**Note**

Sets up sequencer options for a conversion sequence. This function should be used to setup the selected channels for the sequence, the sequencer trigger, the trigger polarity, synchronization bypass, priority, and mode. All options are passed to the functions as a OR'ed list of values. This function will disable/clear the sequencer start/burst/single step/enable if they are enabled.
Select the channels by OR'ing in one or more ADC_SEQ_CTRL_CHANSEL(ch) values.
Select the hardware trigger by OR'ing in one ADC_SEQ_CTRL_HWTRIG_∗ value.
Select a positive edge hardware trigger by OR'ing in ADC_SEQ_CTRL_HWTRIG_POLPOS.
Select trigger bypass synchronisation by OR'ing in ADC_SEQ_CTRL_HWTRIG_SYNCBYPASS.
Select ADC single step on trigger/start by OR'ing in ADC_SEQ_CTRL_SINGLESTEP.
Select higher priority conversion on the other sequencer by OR'ing in ADC_SEQ_CTRL_LOWPRIO.
Select end of seqeuence instead of end of conversion interrupt by OR'ing in ADC_SEQ_CTRL_MODE_EOS.
Example for setting up sequencer A (channels 0-2, trigger on high edge of PIO0_2, interrupt on end of sequence):
Chip_ADC_SetupSequencer(LPC_ADC, ADC_SEQA_IDX, ( ADC_SEQ_CTRL_CHANSEL(0) | ADC_SEQ_CTRL_CHANSEL(1) | ADC_SEQ_CTRL_CHANSEL(2) | ADC_SEQ_CTRL_HWTRIG_PIO0_2 | ADC_SEQ_CTRL_HWTRIG_POLPOS | ADC_SEQ_CTRL_MODE_EOS));

Definition at line 335 of file adc_5411x.h.

#### 6.4.4.27 __STATIC_INLINE void Chip_ADC_StartBurstSequencer ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex* )

Starts sequencer burst mode.

**Parameters**

| | |
|---:|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to start burst on |

**Returns**

Nothing

**Note**

This function sets the BURST bit for the sequencer to force continuous conversion. Use Chip_ADC_StopBurstSequencer() to stop the ADC burst sequence. START and BURST bits can not be set at the same time, thus, START bit will be cleared.

Definition at line 398 of file adc_5411x.h.

#### 6.4.4.28 __STATIC_INLINE void Chip_ADC_StartSequencer ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex* )

Forces a sequencer trigger event (software trigger of ADC)

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to start |

**Returns**

Nothing

**Note**

This function sets the START bit for the sequencer to force a single conversion sequence or a single step conversion. START and BURST bits can not be set at the same time, thus, BURST bit will be cleared.

Definition at line 382 of file adc_5411x.h.

#### 6.4.4.29 __STATIC_INLINE void Chip_ADC_StopBurstSequencer ( LPC_ADC_T ∗ *pADC,* ADC_SEQ_IDX_T *seqIndex* )

Stops sequencer burst mode.

**Parameters**

| | |
|---|---|
| *pADC* | : The base of ADC peripheral on the chip |
| *seqIndex* | : Sequencer to stop burst on |

**Returns**

Nothing

Definition at line 411 of file adc_5411x.h.

## 6.5 CHIP: CHIP_LPC5411X family IRQ vector names and mapped NVIC IRQ numbers

These are the interrupt vector names and the NVIC IRQ number mapping used for the LPC5411X family. You can override any WEAK function with a function in your application. If you do not create a function with one of these names in your application, the vector will be routed to a default handler (dead loop).

**Use example**

```
/* Function created that handles PININT3 interrupt */
void PIN_INT3_IRQHandler(void)
{
 Chip_PININT_ClearIntStatus(LPC_PININT, PININTCH3);
 Board_LED_Toggle(0);
}

/* Enable IRQ for PININT3 */
NVIC_EnableIRQ(PIN_INT3_IRQn);
```

Also see CHIP_5411X: LPC5411X M0 core peripheral interrupt numbers Also see CHIP_5411X: LPC5411X M4 core peripheral interrupt numbers

**MX Core default handler names**
void ResetISR(void); (Reset_IRQn)
WEAK void NMI_Handler(void); (NonMaskableInt_IRQn)
WEAK void HardFault_Handler(void); (HardFault_IRQn)
WEAK void SVC_Handler(void); (SVCall_IRQn)
WEAK void PendSV_Handler(void); (PendSV_IRQn)
WEAK void SysTick_Handler(void); (SysTick_IRQn)

**Peripheral vector names and mapped IRQs**
WEAK void WDT_IRQHandler(void); (WDT_IRQn)
WEAK void BOD_IRQHandler(void); (BOD_IRQn)
WEAK void DMA_IRQHandler(void); (DMA_IRQn)
WEAK void GINT0_IRQHandler(void); (GINT0_IRQn)
WEAK void PIN_INT0_IRQHandler(void); (PIN_INT0_IRQn)
WEAK void PIN_INT1_IRQHandler(void); (PIN_INT1_IRQn)
WEAK void PIN_INT2_IRQHandler(void); (PIN_INT2_IRQn)
WEAK void PIN_INT3_IRQHandler(void); (PIN_INT3_IRQn)
WEAK void UTICK_IRQHandler(void); (UTICK_IRQn)
WEAK void MRT_IRQHandler(void); (MRT_IRQn)
WEAK void CT32B0_IRQHandler(void); (CT32B0_IRQn)
WEAK void CT32B1_IRQHandler(void); (CT32B1_IRQn)
WEAK void CT32B2_IRQHandler(void); (CT32B2_IRQn)
WEAK void CT32B3_IRQHandler(void); (CT32B3_IRQn)
WEAK void CT32B4_IRQHandler(void); (CT32B4_IRQn)
WEAK void SCT0_IRQHandler(void); (SCT0_IRQn)
WEAK void UART0_IRQHandler(void); (UART0_IRQn)
WEAK void UART1_IRQHandler(void); (UART1_IRQn)
WEAK void UART2_IRQHandler(void); (UART2_IRQn)
WEAK void UART3_IRQHandler(void); (UART3_IRQn)
WEAK void I2C0_IRQHandler(void); (I2C0_IRQn)
WEAK void I2C1_IRQHandler(void); (I2C1_IRQn)
WEAK void I2C2_IRQHandler(void); (I2C2_IRQn)
WEAK void SPI0_IRQHandler(void); (SPI0_IRQn)
WEAK void SPI1_IRQHandler(void); (SPI1_IRQn)
WEAK void ADC_SEQA_IRQHandler(void); (ADC_SEQA_IRQn)
WEAK void ADC_SEQB_IRQHandler(void); (ADC_SEQB_IRQn)
WEAK void ADC_THCMP_IRQHandler(void); (ADC_THCMP_IRQn)
WEAK void RTC_IRQHandler(void); (RTC_IRQn)

WEAK void IOH_IRQHandler(void); (IOH_IRQn)
WEAK void MAILBOX_IRQHandler(void); (MAILBOX_IRQn)
WEAK void GINT1_IRQHandler(void); (GINT1_IRQn)
WEAK void PIN_INT4_IRQHandler(void); (PIN_INT4_IRQn)
WEAK void PIN_INT5_IRQHandler(void); (PIN_INT5_IRQn)
WEAK void PIN_INT6_IRQHandler(void); (PIN_INT6_IRQn)
WEAK void PIN_INT7_IRQHandler(void); (PIN_INT7_IRQn)
WEAK void RIT_IRQHandler(void); (RIT_IRQn)

## 6.6 CHIP: Common Chip ISP/IAP commands and return codes

### 6.6.1 Detailed Description

### Macros

- #define IAP_PREWRRITE_CMD 50
- #define IAP_WRISECTOR_CMD 51
- #define IAP_ERSSECTOR_CMD 52
- #define IAP_BLANK_CHECK_SECTOR_CMD 53
- #define IAP_REPID_CMD 54
- #define IAP_READ_BOOT_CODE_CMD 55
- #define IAP_COMPARE_CMD 56
- #define IAP_REINVOKE_ISP_CMD 57
- #define IAP_READ_UID_CMD 58
- #define IAP_ERASE_PAGE_CMD 59
- #define IAP_EEPROM_WRITE 61
- #define IAP_EEPROM_READ 62
- #define IAP_CMD_SUCCESS 0
- #define IAP_INVALID_COMMAND 1
- #define IAP_SRC_ADDR_ERROR 2
- #define IAP_DST_ADDR_ERROR 3
- #define IAP_SRC_ADDR_NOT_MAPPED 4
- #define IAP_DST_ADDR_NOT_MAPPED 5
- #define IAP_COUNT_ERROR 6
- #define IAP_INVALID_SECTOR 7
- #define IAP_SECTOR_NOT_BLANK 8
- #define IAP_SECTOR_NOT_PREPARED 9
- #define IAP_COMPARE_ERROR 10
- #define IAP_BUSY 11
- #define IAP_PARAM_ERROR 12
- #define IAP_ADDR_ERROR 13
- #define IAP_ADDR_NOT_MAPPED 14
- #define IAP_CMD_LOCKED 15
- #define IAP_INVALID_CODE 16
- #define IAP_INVALID_BAUD_RATE 17
- #define IAP_INVALID_STOP_BIT 18
- #define IAP_CRP_ENABLED 19

### Typedefs

- typedef void(∗ IAP_ENTRY_T) (unsigned int[5], unsigned int[4])

### Functions

- uint8_t Chip_IAP_PreSectorForReadWrite (uint32_t strSector, uint32_t endSector)

  *Prepare sector for write operation.*
- uint8_t Chip_IAP_CopyRamToFlash (uint32_t dstAdd, uint32_t ∗srcAdd, uint32_t byteswrt)

  *Copy RAM to flash.*
- uint8_t Chip_IAP_EraseSector (uint32_t strSector, uint32_t endSector)

  *Erase sector.*
- uint8_t Chip_IAP_BlankCheckSector (uint32_t strSector, uint32_t endSector)

  *Blank check a sector or multiples sector of on-chip flash memory.*

- uint32_t Chip_IAP_ReadPID (void)

  *Read part identification number.*
- uint8_t Chip_IAP_ReadBootCode (void)

  *Read boot code version number.*
- uint8_t Chip_IAP_Compare (uint32_t dstAdd, uint32_t srcAdd, uint32_t bytescmp)

  *Compare the memory contents at two locations.*
- uint8_t Chip_IAP_ReinvokeISP (void)

  *IAP reinvoke ISP to invoke the bootloader in ISP mode.*
- uint32_t Chip_IAP_ReadUID (void)

  *Read the unique ID.*
- uint8_t Chip_IAP_ErasePage (uint32_t strPage, uint32_t endPage)

  *Erase a page or multiple papers of on-chip flash memory.*

## 6.6.2 Macro Definition Documentation

### 6.6.2.1 #define IAP_ADDR_ERROR 13

Address is not on word boundary

Definition at line 72 of file iap.h.

### 6.6.2.2 #define IAP_ADDR_NOT_MAPPED 14

Address is not mapped in the memory map

Definition at line 73 of file iap.h.

### 6.6.2.3 #define IAP_BLANK_CHECK_SECTOR_CMD 53

Blank check sector

Definition at line 48 of file iap.h.

### 6.6.2.4 #define IAP_BUSY 11

Flash programming hardware interface is busy

Definition at line 70 of file iap.h.

### 6.6.2.5 #define IAP_CMD_LOCKED 15

Command is locked

Definition at line 74 of file iap.h.

### 6.6.2.6 #define IAP_CMD_SUCCESS 0

Command is executed successfully

Definition at line 59 of file iap.h.

### 6.6.2.7 #define IAP_COMPARE_CMD 56

Compare two RAM address locations

Definition at line 51 of file iap.h.

#### 6.6.2.8 #define IAP_COMPARE_ERROR 10

Source and destination data not equal

Definition at line 69 of file iap.h.

#### 6.6.2.9 #define IAP_COUNT_ERROR 6

Byte count is not multiple of 4 or is not a permitted value

Definition at line 65 of file iap.h.

#### 6.6.2.10 #define IAP_CRP_ENABLED 19

Code read protection enabled

Definition at line 78 of file iap.h.

#### 6.6.2.11 #define IAP_DST_ADDR_ERROR 3

Destination address is not on a correct boundary

Definition at line 62 of file iap.h.

#### 6.6.2.12 #define IAP_DST_ADDR_NOT_MAPPED 5

Destination address is not mapped in the memory map

Definition at line 64 of file iap.h.

#### 6.6.2.13 #define IAP_EEPROM_READ 62

EEPROM READ command

Definition at line 56 of file iap.h.

#### 6.6.2.14 #define IAP_EEPROM_WRITE 61

EEPROM Write command

Definition at line 55 of file iap.h.

#### 6.6.2.15 #define IAP_ERASE_PAGE_CMD 59

Erase page

Definition at line 54 of file iap.h.

#### 6.6.2.16 #define IAP_ERSSECTOR_CMD 52

Erase Sector command

Definition at line 47 of file iap.h.

#### 6.6.2.17 #define IAP_INVALID_BAUD_RATE 17

Invalid baud rate setting

Definition at line 76 of file iap.h.

#### 6.6.2.18 #define IAP_INVALID_CODE 16

Unlock code is invalid

Definition at line 75 of file iap.h.

#### 6.6.2.19 #define IAP_INVALID_COMMAND 1

Invalid command

Definition at line 60 of file iap.h.

#### 6.6.2.20 #define IAP_INVALID_SECTOR 7

Sector number is invalid or end sector number is greater than start sector number

Definition at line 66 of file iap.h.

#### 6.6.2.21 #define IAP_INVALID_STOP_BIT 18

Invalid stop bit setting

Definition at line 77 of file iap.h.

#### 6.6.2.22 #define IAP_PARAM_ERROR 12

nsufficient number of parameters or invalid parameter

Definition at line 71 of file iap.h.

#### 6.6.2.23 #define IAP_PREWRRITE_CMD 50

Prepare sector for write operation command

Definition at line 45 of file iap.h.

#### 6.6.2.24 #define IAP_READ_BOOT_CODE_CMD 55

Read Boot code version

Definition at line 50 of file iap.h.

#### 6.6.2.25 #define IAP_READ_UID_CMD 58

Read UID

Definition at line 53 of file iap.h.

#### 6.6.2.26 #define IAP_REINVOKE_ISP_CMD 57

Reinvoke ISP

Definition at line 52 of file iap.h.

#### 6.6.2.27 #define IAP_REPID_CMD 54

Read PartID command

Definition at line 49 of file iap.h.

#### 6.6.2.28 #define IAP_SECTOR_NOT_BLANK 8

Sector is not blank

Definition at line 67 of file iap.h.

#### 6.6.2.29 #define IAP_SECTOR_NOT_PREPARED 9

Command to prepare sector for write operation was not executed

Definition at line 68 of file iap.h.

#### 6.6.2.30 #define IAP_SRC_ADDR_ERROR 2

Source address is not on word boundary

Definition at line 61 of file iap.h.

#### 6.6.2.31 #define IAP_SRC_ADDR_NOT_MAPPED 4

Source address is not mapped in the memory map

Definition at line 63 of file iap.h.

#### 6.6.2.32 #define IAP_WRISECTOR_CMD 51

Write Sector command

Definition at line 46 of file iap.h.

### 6.6.3 Typedef Documentation

#### 6.6.3.1 typedef void(∗ IAP_ENTRY_T) (unsigned int[5], unsigned int[4])

Definition at line 81 of file iap.h.

### 6.6.4 Function Documentation

#### 6.6.4.1 uint8_t Chip_IAP_BlankCheckSector ( uint32_t *strSector,* uint32_t *endSector* )

Blank check a sector or multiples sector of on-chip flash memory.

**Parameters**

| | |
|---|---|
| *strSector* | : Start sector number |
| *endSector* | : End sector number |

**Returns**

Offset of the first non blank word location if the status code is SECTOR_NOT_BLANK

**Note**

The end sector must be greater than or equal to start sector number

Definition at line 93 of file iap.c.

#### 6.6.4.2 uint8_t Chip_IAP_Compare ( uint32_t *dstAdd,* uint32_t *srcAdd,* uint32_t *bytescmp* )

Compare the memory contents at two locations.

**Parameters**

| | |
|---|---|
| *dstAdd* | : Destination of the RAM address of data bytes to be compared |
| *srcAdd* | : Source of the RAM address of data bytes to be compared |
| *bytescmp* | : Number of bytes to be compared |

**Returns**

Offset of the first mismatch of the status code is COMPARE_ERROR

**Note**

The addresses should be a word boundary and number of bytes should be a multiply of 4

Definition at line 128 of file iap.c.

#### 6.6.4.3 uint8_t Chip_IAP_CopyRamToFlash ( uint32_t *dstAdd,* uint32_t ∗ *srcAdd,* uint32_t *byteswrt* )

Copy RAM to flash.

**Parameters**

| | |
|---|---|
| *dstAdd* | : Destination flash address where data bytes are to be written |
| *srcAdd* | : Source flash address where data bytes are to be read |
| *byteswrt* | : Number of bytes to be written |

**Returns**

Status code to indicate the command is executed successfully or not

**Note**

The addresses should be a 256 byte boundary and the number of bytes should be 256 | 512 | 1024 | 4096

Definition at line 64 of file iap.c.

#### 6.6.4.4 uint8_t Chip_IAP_ErasePage ( uint32_t *strPage,* uint32_t *endPage* )

Erase a page or multiple papers of on-chip flash memory.

**Parameters**

| | |
|---|---|
| *strPage* | : Start page number |
| *endPage* | : End page number |

**Returns**

Status code to indicate the command is executed successfully or not

**Note**

The page number must be greater than or equal to start page number

Definition at line 164 of file iap.c.

### 6.6.4.5 uint8_t Chip_IAP_EraseSector ( uint32_t *strSector,* uint32_t *endSector* )

Erase sector.

**Parameters**

| | |
|---|---|
| *strSector* | : Start sector number |
| *endSector* | : End sector number |

**Returns**

Status code to indicate the command is executed successfully or not

**Note**

The end sector must be greater than or equal to start sector number

Definition at line 79 of file iap.c.

### 6.6.4.6 uint8_t Chip_IAP_PreSectorForReadWrite ( uint32_t *strSector,* uint32_t *endSector* )

Prepare sector for write operation.

**Parameters**

| | |
|---|---|
| *strSector* | : Start sector number |
| *endSector* | : End sector number |

**Returns**

Status code to indicate the command is executed successfully or not

**Note**

This command must be executed before executing "Copy RAM to flash" or "Erase Sector" command. The end sector must be greater than or equal to start sector number

Definition at line 51 of file iap.c.

#### 6.6.4.7 uint8_t Chip_IAP_ReadBootCode ( void )

Read boot code version number.

**Returns**

Boot code version number

Definition at line 117 of file iap.c.

#### 6.6.4.8 uint32_t Chip_IAP_ReadPID ( void )

Read part identification number.

**Returns**

Part identification number

Definition at line 106 of file iap.c.

#### 6.6.4.9 uint32_t Chip_IAP_ReadUID ( void )

Read the unique ID.

**Returns**

Status code to indicate the command is executed successfully or not

Definition at line 153 of file iap.c.

#### 6.6.4.10 uint8_t Chip_IAP_ReinvokeISP ( void )

IAP reinvoke ISP to invoke the bootloader in ISP mode.

**Returns**

none

Definition at line 142 of file iap.c.

## 6.7 CHIP: FPU initialization

### 6.7.1 Detailed Description

Cortex FPU initialization

### Functions

- void fpuInit (void)

  *Early initialization of the FPU.*

### 6.7.2 Function Documentation

#### 6.7.2.1 void fpuInit ( void )

Early initialization of the FPU.

**Returns**

Nothing

# 6.8 CHIP: LPC Common Types

## 6.8.1 Detailed Description

### Modules

- LPC Public Macros
- LPC Public Types

## 6.9 CHIP: LPC5410x family CMSIS include files

IRQ vector names and NVIC vector mapping for the LPC5410x device

## 6.10 CHIP: LPC5411X 32-bit Timer driver

### 6.10.1 Detailed Description

#### Data Structures

- struct LPC_TIMER_T

  *32-bit Standard timer register block structure*

#### Macros

- #define TIMER_IR_CLR(n) _BIT(n)
- #define TIMER_MATCH_INT(n) (_BIT((n) & 0x0F))
- #define TIMER_CAP_INT(n) (_BIT((((n) & 0x0F) + 4)))
- #define TIMER_ENABLE ((uint32_t) (1 << 0))
- #define TIMER_RESET ((uint32_t) (1 << 1))
- #define TIMER_CTRL_MASK ((uint32_t) 0x03)
- #define TIMER_INT_ON_MATCH(n) (_BIT(((n) ∗ 3)))
- #define TIMER_RESET_ON_MATCH(n) (_BIT((((n) ∗ 3) + 1)))
- #define TIMER_STOP_ON_MATCH(n) (_BIT((((n) ∗ 3) + 2)))
- #define TIMER_MCR_MASK ((uint32_t) 0x0FFF)
- #define TIMER_CAP_RISING(n) (_BIT(((n) ∗ 3)))
- #define TIMER_CAP_FALLING(n) (_BIT((((n) ∗ 3) + 1)))
- #define TIMER_INT_ON_CAP(n) (_BIT((((n) ∗ 3) + 2)))
- #define TIMER_CCR_MASK ((uint32_t) 0x0FFF)
- #define TIMER_EMR_MASK ((uint32_t) 0x0FFF)
- #define TIMER_CTCR_MASK ((uint32_t) 0x0F)

#### Enumerations

- enum TIMER_PIN_MATCH_STATE_T { TIMER_EXTMATCH_DO_NOTHING = 0, TIMER_EXTMATCH_CLEAR = 1, TIMER_EXTMATCH_SET = 2, TIMER_EXTMATCH_TOGGLE = 3 }

  *Standard timer initial match pin state and change state.*
- enum TIMER_CAP_SRC_STATE_T { TIMER_CAPSRC_RISING_PCLK = 0, TIMER_CAPSRC_RISING_CAPN = 1, TIMER_CAPSRC_FALLING_CAPN = 2, TIMER_CAPSRC_BOTH_CAPN = 3 }

  *Standard timer clock and edge for count source.*

#### Functions

- __STATIC_INLINE void Chip_TIMER_Init (LPC_TIMER_T ∗pTMR)

  *Initialize a timer.*
- __STATIC_INLINE void Chip_TIMER_DeInit (LPC_TIMER_T ∗pTMR)

  *Shutdown a timer.*
- __STATIC_INLINE bool Chip_TIMER_MatchPending (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Determine if a match interrupt is pending.*
- __STATIC_INLINE bool Chip_TIMER_CapturePending (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Determine if a capture interrupt is pending.*
- __STATIC_INLINE void Chip_TIMER_ClearMatch (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Clears a (pending) match interrupt.*
- __STATIC_INLINE void Chip_TIMER_ClearCapture (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Clears a (pending) capture interrupt.*

- __STATIC_INLINE void Chip_TIMER_Enable (LPC_TIMER_T ∗pTMR)

  *Enables the timer (starts count)*
- __STATIC_INLINE void Chip_TIMER_Disable (LPC_TIMER_T ∗pTMR)

  *Disables the timer (stops count)*
- __STATIC_INLINE uint32_t Chip_TIMER_ReadCount (LPC_TIMER_T ∗pTMR)

  *Returns the current timer count.*
- __STATIC_INLINE uint32_t Chip_TIMER_ReadPrescale (LPC_TIMER_T ∗pTMR)

  *Returns the current prescale count.*
- __STATIC_INLINE void Chip_TIMER_PrescaleSet (LPC_TIMER_T ∗pTMR, uint32_t prescale)

  *Sets the prescaler value.*
- __STATIC_INLINE void Chip_TIMER_SetMatch (LPC_TIMER_T ∗pTMR, int8_t matchnum, uint32_t matchval)

  *Sets a timer match value.*
- __STATIC_INLINE uint32_t Chip_TIMER_ReadCapture (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Reads a capture register.*
- void Chip_TIMER_Reset (LPC_TIMER_T ∗pTMR)

  *Resets the timer terminal and prescale counts to 0.*
- __STATIC_INLINE void Chip_TIMER_MatchEnableInt (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Enables a match interrupt that fires when the terminal count matches the match counter value.*
- __STATIC_INLINE void Chip_TIMER_MatchDisableInt (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Disables a match interrupt for a match counter.*
- __STATIC_INLINE void Chip_TIMER_ResetOnMatchEnable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *For the specific match counter, enables reset of the terminal count register when a match occurs.*
- __STATIC_INLINE void Chip_TIMER_ResetOnMatchDisable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *For the specific match counter, disables reset of the terminal count register when a match occurs.*
- __STATIC_INLINE void Chip_TIMER_StopOnMatchEnable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Enable a match timer to stop the terminal count when a match count equals the terminal count.*
- __STATIC_INLINE void Chip_TIMER_StopOnMatchDisable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Disable stop on match for a match timer. Disables a match timer to stop the terminal count when a match count equals the terminal count.*
- __STATIC_INLINE void Chip_TIMER_CaptureRisingEdgeEnable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Enables capture on on rising edge of selected CAP signal for the selected capture register, enables the selected CAPn.capnum signal to load the capture register with the terminal coount on a rising edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureRisingEdgeDisable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Disables capture on on rising edge of selected CAP signal. For the selected capture register, disables the selected CAPn.capnum signal to load the capture register with the terminal coount on a rising edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureFallingEdgeEnable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Enables capture on on falling edge of selected CAP signal. For the selected capture register, enables the selected CAPn.capnum signal to load the capture register with the terminal coount on a falling edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureFallingEdgeDisable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Disables capture on on falling edge of selected CAP signal. For the selected capture register, disables the selected CAPn.capnum signal to load the capture register with the terminal coount on a falling edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureEnableInt (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Enables interrupt on capture of selected CAP signal. For the selected capture register, an interrupt will be generated when the enabled rising or falling edge on CAPn.capnum is detected.*
- __STATIC_INLINE void Chip_TIMER_CaptureDisableInt (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Disables interrupt on capture of selected CAP signal.*
- void Chip_TIMER_ExtMatchControlSet (LPC_TIMER_T ∗pTMR, int8_t initial_state, TIMER_PIN_MATCH_STATE_T matchState, int8_t matchnum)

  *Sets external match control (MATn.matchnum) pin control. For the pin selected with matchnum, sets the function of the pin that occurs on a terminal count match for the match count.*
- __STATIC_INLINE void Chip_TIMER_TIMER_SetCountClockSrc (LPC_TIMER_T ∗pTMR, TIMER_CAP_SRC_STATE_T capSrc, int8_t capnum)

  *Sets timer count source and edge with the selected passed from CapSrc. If CapSrc selected a CAPn pin, select the specific CAPn pin with the capnum value.*

## 6.10.2 Macro Definition Documentation

### 6.10.2.1 #define TIMER_CAP_FALLING( *n* ) (_BIT((((n) ∗ 3) + 1)))

Bit location for CAP.n on CRx falling edge, n = 0 to 3

Definition at line 90 of file timer_5411x.h.

### 6.10.2.2 #define TIMER_CAP_INT( *n* ) (_BIT((((n) & 0x0F) + 4)))

Macro for getting a capture event interrupt bit

Definition at line 69 of file timer_5411x.h.

### 6.10.2.3 #define TIMER_CAP_RISING( *n* ) (_BIT(((n) ∗ 3)))

Bit location for CAP.n on CRx rising edge, n = 0 to 3

Definition at line 88 of file timer_5411x.h.

### 6.10.2.4 #define TIMER_CCR_MASK ((uint32_t) 0x0FFF)

Capture Control register Mask

Definition at line 94 of file timer_5411x.h.

### 6.10.2.5 #define TIMER_CTCR_MASK ((uint32_t) 0x0F)

Counter Control register Mask

Definition at line 98 of file timer_5411x.h.

### 6.10.2.6 #define TIMER_CTRL_MASK ((uint32_t) 0x03)

Timer Control register Mask

Definition at line 76 of file timer_5411x.h.

### 6.10.2.7 #define TIMER_EMR_MASK ((uint32_t) 0x0FFF)

External Match register Mask

Definition at line 96 of file timer_5411x.h.

### 6.10.2.8 #define TIMER_ENABLE ((uint32_t) (1 << 0))

Timer/counter enable bit

Definition at line 72 of file timer_5411x.h.

### 6.10.2.9 #define TIMER_INT_ON_CAP( *n* ) (_BIT((((n) ∗ 3) + 2)))

Bit location for CAP.n on CRx interrupt enable, n = 0 to 3

Definition at line 92 of file timer_5411x.h.

#### 6.10.2.10 #define TIMER_INT_ON_MATCH( *n* ) (_BIT(((n) $*$ 3)))

Bit location for interrupt on MRx match, n = 0 to 3

Definition at line 79 of file timer_5411x.h.

#### 6.10.2.11 #define TIMER_IR_CLR( *n* ) _BIT(n)

Macro to clear interrupt pending

Definition at line 64 of file timer_5411x.h.

#### 6.10.2.12 #define TIMER_MATCH_INT( *n* ) (_BIT((n) & 0x0F))

Macro for getting a timer match interrupt bit

Definition at line 67 of file timer_5411x.h.

#### 6.10.2.13 #define TIMER_MCR_MASK ((uint32_t) 0x0FFF)

Match Control register Mask

Definition at line 85 of file timer_5411x.h.

#### 6.10.2.14 #define TIMER_RESET ((uint32_t) (1 $<<$ 1))

Timer/counter reset bit

Definition at line 74 of file timer_5411x.h.

#### 6.10.2.15 #define TIMER_RESET_ON_MATCH( *n* ) (_BIT((((n) $*$ 3) + 1)))

Bit location for reset on MRx match, n = 0 to 3

Definition at line 81 of file timer_5411x.h.

#### 6.10.2.16 #define TIMER_STOP_ON_MATCH( *n* ) (_BIT((((n) $*$ 3) + 2)))

Bit location for stop on MRx match, n = 0 to 3

Definition at line 83 of file timer_5411x.h.

### 6.10.3 Enumeration Type Documentation

#### 6.10.3.1 enum TIMER_CAP_SRC_STATE_T

Standard timer clock and edge for count source.

**Enumerator**

***TIMER_CAPSRC_RISING_PCLK*** Timer ticks on PCLK rising edge
***TIMER_CAPSRC_RISING_CAPN*** Timer ticks on CAPn.x rising edge
***TIMER_CAPSRC_FALLING_CAPN*** Timer ticks on CAPn.x falling edge
***TIMER_CAPSRC_BOTH_CAPN*** Timer ticks on CAPn.x both edges

Definition at line 464 of file timer_5411x.h.

#### 6.10.3.2 enum TIMER_PIN_MATCH_STATE_T

Standard timer initial match pin state and change state.

**Enumerator**

***TIMER_EXTMATCH_DO_NOTHING*** Timer match state does nothing on match pin

***TIMER_EXTMATCH_CLEAR*** Timer match state sets match pin low

***TIMER_EXTMATCH_SET*** Timer match state sets match pin high

***TIMER_EXTMATCH_TOGGLE*** Timer match state toggles match pin

Definition at line 439 of file timer_5411x.h.

### 6.10.4 Function Documentation

#### 6.10.4.1 __STATIC_INLINE void Chip_TIMER_CaptureDisableInt ( LPC_TIMER_T ∗ *pTMR,* int8_t *capnum* )

Disables interrupt on capture of selected CAP signal.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture signal/register to use |

**Returns**

Nothing

Definition at line 431 of file timer_5411x.h.

#### 6.10.4.2 __STATIC_INLINE void Chip_TIMER_CaptureEnableInt ( LPC_TIMER_T ∗ *pTMR,* int8_t *capnum* )

Enables interrupt on capture of selected CAP signal. For the selected capture register, an interrupt will be generated when the enabled rising or falling edge on CAPn.capnum is detected.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture signal/register to use |

**Returns**

Nothing

Definition at line 420 of file timer_5411x.h.

#### 6.10.4.3 __STATIC_INLINE void Chip_TIMER_CaptureFallingEdgeDisable ( LPC_TIMER_T ∗ *pTMR,* int8_t *capnum* )

Disables capture on on falling edge of selected CAP signal. For the selected capture register, disables the selected CAPn.capnum signal to load the capture register with the terminal coount on a falling edge.

**Parameters**

| | |
|---:|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture signal/register to use |

**Returns**

Nothing

Definition at line 407 of file timer_5411x.h.

#### 6.10.4.4 __STATIC_INLINE void Chip_TIMER_CaptureFallingEdgeEnable ( LPC_TIMER_T ∗ *pTMR,* int8_t *capnum* )

Enables capture on on falling edge of selected CAP signal. For the selected capture register, enables the selected CAPn.capnum signal to load the capture register with the terminal coount on a falling edge.

**Parameters**

| | |
|---:|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture signal/register to use |

**Returns**

Nothing

Definition at line 394 of file timer_5411x.h.

#### 6.10.4.5 __STATIC_INLINE bool Chip_TIMER_CapturePending ( LPC_TIMER_T ∗ *pTMR,* int8_t *capnum* )

Determine if a capture interrupt is pending.

**Parameters**

| | |
|---:|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture interrupt number to check |

**Returns**

false if the interrupt is not pending, otherwise true

**Note**

Determine if the capture interrupt for the passed capture pin is pending.

Definition at line 174 of file timer_5411x.h.

#### 6.10.4.6 __STATIC_INLINE void Chip_TIMER_CaptureRisingEdgeDisable ( LPC_TIMER_T ∗ *pTMR,* int8_t *capnum* )

Disables capture on on rising edge of selected CAP signal. For the selected capture register, disables the selected CAPn.capnum signal to load the capture register with the terminal coount on a rising edge.

**Parameters**

| | |
|---:|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture signal/register to use |

**Returns**

Nothing

Definition at line 381 of file timer_5411x.h.

#### 6.10.4.7 __STATIC_INLINE void Chip_TIMER_CaptureRisingEdgeEnable ( LPC_TIMER_T * *pTMR,* int8_t *capnum* )

Enables capture on on rising edge of selected CAP signal for the selected capture register, enables the selected CAPn.capnum signal to load the capture register with the terminal coount on a rising edge.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture signal/register to use |

**Returns**

Nothing

Definition at line 368 of file timer_5411x.h.

#### 6.10.4.8 __STATIC_INLINE void Chip_TIMER_ClearCapture ( LPC_TIMER_T ∗ *pTMR*, int8_t *capnum* )

Clears a (pending) capture interrupt.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture interrupt number to clear |

**Returns**

Nothing

**Note**

Clears a pending timer capture interrupt.

Definition at line 198 of file timer_5411x.h.

#### 6.10.4.9 __STATIC_INLINE void Chip_TIMER_ClearMatch ( LPC_TIMER_T ∗ *pTMR*, int8_t *matchnum* )

Clears a (pending) match interrupt.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match interrupt number to clear |

**Returns**

Nothing

**Note**

Clears a pending timer match interrupt.

Definition at line 186 of file timer_5411x.h.

#### 6.10.4.10 __STATIC_INLINE void Chip_TIMER_DeInit ( LPC_TIMER_T ∗ *pTMR* )

Shutdown a timer.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |

**Returns**

Nothing

Definition at line 134 of file timer_5411x.h.

#### 6.10.4.11 __STATIC_INLINE void Chip_TIMER_Disable ( LPC_TIMER_T ∗ *pTMR* )

Disables the timer (stops count)

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |

**Returns**

Nothing

**Note**

Disables the timer to stop counting.

Definition at line 220 of file timer_5411x.h.

#### 6.10.4.12 __STATIC_INLINE void Chip_TIMER_Enable ( LPC_TIMER_T ∗ *pTMR* )

Enables the timer (starts count)

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |

**Returns**

Nothing

**Note**

Enables the timer to start counting.

Definition at line 209 of file timer_5411x.h.

#### 6.10.4.13 void Chip_TIMER_ExtMatchControlSet ( LPC_TIMER_T ∗ *pTMR,* int8_t *initial_state,* TIMER_PIN_MATCH_STATE_T *matchState,* int8_t *matchnum* )

Sets external match control (MATn.matchnum) pin control. For the pin selected with matchnum, sets the function of the pin that occurs on a terminal count match for the match count.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *initial_state* | : Initial state of the pin, high(1) or low(0) |
| *matchState* | : Selects the match state for the pin |
| *matchnum* | : MATn.matchnum signal to use |

**Returns**

Nothing

**Note**

For the pin selected with matchnum, sets the function of the pin that occurs on a terminal count match for the match count.

Definition at line 72 of file timer_5411x.c.

#### 6.10.4.14 __STATIC_INLINE void Chip_TIMER_Init ( LPC_TIMER_T ∗ *pTMR* )

Initialize a timer.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |

**Returns**

Nothing

Definition at line 105 of file timer_5411x.h.

#### 6.10.4.15 __STATIC_INLINE void Chip_TIMER_MatchDisableInt ( LPC_TIMER_T ∗ *pTMR*, int8_t *matchnum* )

Disables a match interrupt for a match counter.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match timer, 0 to 3 |

**Returns**

Nothing

Definition at line 309 of file timer_5411x.h.

#### 6.10.4.16 __STATIC_INLINE void Chip_TIMER_MatchEnableInt ( LPC_TIMER_T ∗ *pTMR*, int8_t *matchnum* )

Enables a match interrupt that fires when the terminal count matches the match counter value.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |

| | |
|---|---|
| *matchnum* | : Match timer, 0 to 3 |

**Returns**

Nothing

Definition at line 298 of file timer_5411x.h.

#### 6.10.4.17 __STATIC_INLINE bool Chip_TIMER_MatchPending ( LPC_TIMER_T ∗ *pTMR,* int8_t *matchnum* )

Determine if a match interrupt is pending.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match interrupt number to check |

**Returns**

false if the interrupt is not pending, otherwise true

**Note**

Determine if the match interrupt for the passed timer and match counter is pending.

Definition at line 161 of file timer_5411x.h.

#### 6.10.4.18 __STATIC_INLINE void Chip_TIMER_PrescaleSet ( LPC_TIMER_T ∗ *pTMR,* uint32_t *prescale* )

Sets the prescaler value.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *prescale* | : Prescale value to set the prescale register to |

**Returns**

Nothing

**Note**

Sets the prescale count value.

Definition at line 254 of file timer_5411x.h.

#### 6.10.4.19 __STATIC_INLINE uint32_t Chip_TIMER_ReadCapture ( LPC_TIMER_T ∗ *pTMR,* int8_t *capnum* )

Reads a capture register.

**Parameters**

| | |
|---:|---|
| *pTMR* | : Pointer to timer IP register address |
| *capnum* | : Capture register to read |

**Returns**

The selected capture register value

**Note**

Returns the selected capture register value.

Definition at line 279 of file timer_5411x.h.

#### 6.10.4.20 __STATIC_INLINE uint32_t Chip_TIMER_ReadCount ( LPC_TIMER_T ∗ *pTMR* )

Returns the current timer count.

**Parameters**

| | |
|---:|---|
| *pTMR* | : Pointer to timer IP register address |

**Returns**

Current timer terminal count value

**Note**

Returns the current timer terminal count.

Definition at line 231 of file timer_5411x.h.

#### 6.10.4.21 __STATIC_INLINE uint32_t Chip_TIMER_ReadPrescale ( LPC_TIMER_T ∗ *pTMR* )

Returns the current prescale count.

**Parameters**

| | |
|---:|---|
| *pTMR* | : Pointer to timer IP register address |

**Returns**

Current timer prescale count value

**Note**

Returns the current prescale count.

Definition at line 242 of file timer_5411x.h.

#### 6.10.4.22 void Chip_TIMER_Reset ( LPC_TIMER_T ∗ *pTMR* )

Resets the timer terminal and prescale counts to 0.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |

**Returns**

Nothing

Definition at line 52 of file timer_5411x.c.

#### 6.10.4.23 __STATIC_INLINE void Chip_TIMER_ResetOnMatchDisable ( LPC_TIMER_T ∗ *pTMR,* int8_t *matchnum* )

For the specific match counter, disables reset of the terminal count register when a match occurs.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match timer, 0 to 3 |

**Returns**

Nothing

Definition at line 331 of file timer_5411x.h.

#### 6.10.4.24 __STATIC_INLINE void Chip_TIMER_ResetOnMatchEnable ( LPC_TIMER_T ∗ *pTMR,* int8_t *matchnum* )

For the specific match counter, enables reset of the terminal count register when a match occurs.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match timer, 0 to 3 |

**Returns**

Nothing

Definition at line 320 of file timer_5411x.h.

#### 6.10.4.25 __STATIC_INLINE void Chip_TIMER_SetMatch ( LPC_TIMER_T ∗ *pTMR,* int8_t *matchnum,* uint32_t *matchval* )

Sets a timer match value.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match timer to set match count for |
| *matchval* | : Match value for the selected match count |

**Returns**

Nothing

**Note**

Sets one of the timer match values.

Definition at line 267 of file timer_5411x.h.

#### 6.10.4.26 __STATIC_INLINE void Chip_TIMER_StopOnMatchDisable ( LPC_TIMER_T ∗ *pTMR,* int8_t *matchnum* )

Disable stop on match for a match timer. Disables a match timer to stop the terminal count when a match count equals the terminal count.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match timer, 0 to 3 |

**Returns**

Nothing

Definition at line 355 of file timer_5411x.h.

#### 6.10.4.27 __STATIC_INLINE void Chip_TIMER_StopOnMatchEnable ( LPC_TIMER_T ∗ *pTMR,* int8_t *matchnum* )

Enable a match timer to stop the terminal count when a match count equals the terminal count.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *matchnum* | : Match timer, 0 to 3 |

**Returns**

Nothing

Definition at line 343 of file timer_5411x.h.

#### 6.10.4.28 __STATIC_INLINE void Chip_TIMER_TIMER_SetCountClockSrc ( LPC_TIMER_T ∗ *pTMR,* TIMER_CAP_SRC_STATE_T *capSrc,* int8_t *capnum* )

Sets timer count source and edge with the selected passed from CapSrc. If CapSrc selected a CAPn pin, select the specific CAPn pin with the capnum value.

**Parameters**

| | |
|---|---|
| *pTMR* | : Pointer to timer IP register address |
| *capSrc* | : timer clock source and edge |
| *capnum* | : CAPn.capnum pin to use (0 - 2) |

**Returns**

Nothing

**Note**

If CapSrc selected a CAPn pin, select the specific CAPn pin with the capnum value.

Definition at line 480 of file timer_5411x.h.

## 6.11 CHIP: LPC5411X CPU multi-core support driver

### 6.11.1 Detailed Description

This driver helps with determine which MCU core the software is running, whether the MCU core is in master or slave mode, and provides functions for master and slave core control.

The functions for the driver are provided as part of the CHIP: LPC5411X Power LIBRARY functions library. For more information on using the LPC5411x LPCopen package with multi-core, see LPC5411X multi-core use in LPCOpen.

### Enumerations

- enum CORESELECT_T { CORESELECT_M0PLUS = 0, CORESELECT_M4 }

### Functions

- __STATIC_INLINE bool Chip_CPU_IsM4Core (void)

  *Determine which MCU this code is running on.*
- void Chip_CPU_SelectMasterCore (CORESELECT_T master, CORESELECT_T ownerPower)

  *Select master core and system power control ownership.*
- bool Chip_CPU_IsMasterCore (void)

  *Determine if this core is a slave or master.*
- void Chip_CPU_CM0Boot (uint32_t ∗coentry, uint32_t ∗costackptr)

  *Setup M0+ boot and reset M0+ core.*
- void Chip_CPU_CM4Boot (uint32_t ∗coentry, uint32_t ∗costackptr)

  *Setup M4 boot and reset M4 core.*

### 6.11.2 Enumeration Type Documentation

#### 6.11.2.1 enum CORESELECT_T

**Enumerator**

***CORESELECT_M0PLUS***

***CORESELECT_M4***

Definition at line 65 of file cpuctrl_5411x.h.

### 6.11.3 Function Documentation

#### 6.11.3.1 void Chip_CPU_CM0Boot ( uint32_t ∗ *coentry*, uint32_t ∗ *costackptr* )

Setup M0+ boot and reset M0+ core.

**Parameters**

| | |
|---|---|
| *coentry* | : Pointer to boot entry point for M0+ core |
| *costackptr* | : Pointer to where stack should be located for M0+ core |

**Returns**

Nothing

**Note**

Will setup boot stack and entry point, enable M0+ clock and then reset M0+ core.

#### 6.11.3.2 void Chip_CPU_CM4Boot ( uint32_t ∗ *coentry,* uint32_t ∗ *costackptr* )

Setup M4 boot and reset M4 core.

**Parameters**

| | |
|---|---|
| *coentry* | : Pointer to boot entry point for M4 core |
| *costackptr* | : Pointer to where stack should be located for M4 core |

**Returns**

Nothing

**Note**

Will setup boot stack and entry point, enable M4 clock and then reset M0+ core.

#### 6.11.3.3 __STATIC_INLINE bool Chip_CPU_IsM4Core ( void )

Determine which MCU this code is running on.

**Returns**

true if executing on the CM4, or false if executing on the CM0+

Definition at line 55 of file cpuctrl_5411x.h.

#### 6.11.3.4 bool Chip_CPU_IsMasterCore ( void )

Determine if this core is a slave or master.

**Returns**

true if this MCU is operating as the master, or false if operating as a slave

#### 6.11.3.5 void Chip_CPU_SelectMasterCore ( CORESELECT_T *master,* CORESELECT_T *ownerPower* )

Select master core and system power control ownership.

**Returns**

Nothing

**Note**

This function can be used to select the master core and which core can powerdown the system. The master core can be re-selected on either the current master or slave core. Power control ownership is used to select which core can place the system in DEEP SLEEP, POWERDOWN, and DEEP POWERDOWN modes. (See Chip_POWER_EnterPowerMode). Note both the master and slave cores can used SLEEP mode, but only the master core can use the other modes.

## 6.12 CHIP: LPC5411X Clock Driver

### 6.12.1 Detailed Description

#### Macros

- #define SYSCON_FRO12MHZ_FREQ (12000000)
- #define SYSCON_FRO48MHZ_FREQ (48000000)
- #define SYSCON_FRO96MHZ_FREQ (96000000)
- #define SYSCON_WDTOSC_FREQ (500000)
- #define SYSCON_RTC_FREQ (32768)
- #define Chip_Clock_GetIntOscRate()  SYSCON_FRO12MHZ_FREQ

#### Enumerations

- enum WDT_OSC_FREQ_T {
  WDT_FREQ_RESERVED, WDT_FREQ_400000, WDT_FREQ_600000, WDT_FREQ_750000,
  WDT_FREQ_900000, WDT_FREQ_1000000, WDT_FREQ_1200000, WDT_FREQ_1300000,
  WDT_FREQ_1400000, WDT_FREQ_1500000, WDT_FREQ_1600000, WDT_FREQ_1700000,
  WDT_FREQ_1800000, WDT_FREQ_1900000, WDT_FREQ_2000000, WDT_FREQ_2050000,
  WDT_FREQ_2100000, WDT_FREQ_2200000, WDT_FREQ_2250000, WDT_FREQ_2300000,
  WDT_FREQ_2400000, WDT_FREQ_2450000, WDT_FREQ_2500000, WDT_FREQ_2600000,
  WDT_FREQ_2650000, WDT_FREQ_2700000, WDT_FREQ_2800000, WDT_FREQ_2850000,
  WDT_FREQ_2900000, WDT_FREQ_2950000, WDT_FREQ_3000000, WDT_FREQ_3050000 }

  *WDT Osc frequency value table.*
- enum CHIP_SYSCON_MAIN_A_CLKSRC_T { SYSCON_MAIN_A_CLKSRC_FRO12MHZ = 0, SYSCON_MAIN_A_CLKSRCA_CLKIN, SYSCON_MAIN_A_CLKSRCA_WDTOSC, SYSCON_MAIN_A_CLKSRCA_FROHF }
- enum CHIP_SYSCON_USBCLKSRC_T { SYSCON_USBCLKSRC_FROHF, SYSCON_USBCLKSRC_PLL, SYSCON_USBCLKSRC_DISABLED = 7 }

  *USB Clock source.*
- enum CHIP_SYSCON_MCLKSRC_T { SYSCON_MCLKSRC_FROHF, SYSCON_MCLKSRC_PLL, SYSCON_MCLKSRC_MCLKIN, SYSCON_MCLKSRC_DISABLED = 7 }

  *MCLK Clock sources.*
- enum CHIP_SYSCON_MAIN_B_CLKSRC_T { SYSCON_MAIN_B_CLKSRC_MAINCLKSELA = 0, SYSCON_MAIN_B_CLKSRC_PLL = 2, SYSCON_MAIN_B_CLKSRC_RTC }
- enum CHIP_SYSCON_CLKOUTSRC_T {
  SYSCON_CLKOUTSRC_MAINCLK = 0, SYSCON_CLKOUTSRC_CLKIN, SYSCON_CLKOUTSRC_WDTOSC, SYSCON_CLKOUTSRC_FROHF,
  SYSCON_CLKOUTSRC_PLL, SYSCON_CLKOUTSRC_FRO12MHZ, SYSCON_CLKOUTSRC_RTC, SYSCON_CLKOUTSRC_DISABLED }
- enum CHIP_SYSCON_CLOCK_T {
  SYSCON_CLOCK_ROM = 1, SYSCON_CLOCK_SRAM1 = 3, SYSCON_CLOCK_SRAM2, SYSCON_CLOCK_SRAMX,
  SYSCON_CLOCK_FLASH = 7, SYSCON_CLOCK_FMC, SYSCON_CLOCK_SPIFI = 10, SYSCON_CLOCK_INPUTMUX,
  SYSCON_CLOCK_IOCON = 13, SYSCON_CLOCK_GPIO0, SYSCON_CLOCK_GPIO1, SYSCON_CLOCK_PINT = 18,
  SYSCON_CLOCK_GINT, SYSCON_CLOCK_DMA, SYSCON_CLOCK_CRC, SYSCON_CLOCK_WWDT,
  SYSCON_CLOCK_RTC, SYSCON_CLOCK_MAILBOX = 26, SYSCON_CLOCK_ADC0, SYSCON_CLOCK_MRT = 32,
  SYSCON_CLOCK_SCT0 = 32 + 2, SYSCON_CLOCK_UTICK = 32 + 10, SYSCON_CLOCK_FLEXCOMM0, SYSCON_CLOCK_FLEXCOMM1,
  SYSCON_CLOCK_FLEXCOMM2, SYSCON_CLOCK_FLEXCOMM3, SYSCON_CLOCK_FLEXCOMM4,

SYSCON_CLOCK_FLEXCOMM5,
SYSCON_CLOCK_FLEXCOMM6, SYSCON_CLOCK_FLEXCOMM7, SYSCON_CLOCK_DMIC, SYSCON_CLOCK_TIMER2 = 32 + 22,
SYSCON_CLOCK_USB = 32 + 25, SYSCON_CLOCK_TIMER0, SYSCON_CLOCK_TIMER1, SYSCON_CLOCK_TIMER3 = 128 + 13,
SYSCON_CLOCK_TIMER4 }
- enum CHIP_SYSCON_FLEXCOMMCLKSELSRC_T {
SYSCON_FLEXCOMMCLKSELSRC_FRO12MHZ = 0, SYSCON_FLEXCOMMCLKSELSRC_FROHF, SYSCON_FLEXCOMMCLKSELSRC_PLL, SYSCON_FLEXCOMMCLKSELSRC_MCLK,
SYSCON_FLEXCOMMCLKSELSRC_FRG, SYSCON_FLEXCOMMCLKSELSRC_NONE = 7 }
- enum CHIP_SYSCON_ADCCLKSELSRC_T { SYSCON_ADCCLKSELSRC_MAINCLK = 0, SYSCON_ADCCLKSELSRC_SYSPLLOUT, SYSCON_ADCCLKSELSRC_FROHF }
- enum CHIP_ASYNC_SYSCON_SRC_T { SYSCON_ASYNC_MAINCLK = 0, SYSCON_ASYNC_FRO12MHZ }
- enum CHIP_SYSCON_MAINCLKSRC_T {
SYSCON_MAINCLKSRC_FRO12MHZ = 0, SYSCON_MAINCLKSRC_CLKIN, SYSCON_MAINCLKSRC_WDTOSC, SYSCON_MAINCLKSRC_FROHF,
SYSCON_MAINCLKSRC_PLLOUT = 6, SYSCON_MAINCLKSRC_RTC }
- enum CHIP_SYSCON_FRGCLKSRC_T {
SYSCON_FRGCLKSRC_MAINCLK, SYSCON_FRGCLKSRC_PLL, SYSCON_FRGCLKSRC_FRO12MHZ, SYSCON_FRGCLKSRC_FROHF,
SYSCON_FRGCLKSRC_NONE = 7 }

  *Fractional Divider clock sources.*

## Functions

- __STATIC_INLINE uint32_t Chip_Clock_GetExtClockInRate (void)

  *Returns the external clock input rate.*
- __STATIC_INLINE uint32_t Chip_Clock_GetRTCOscRate (void)

  *Returns the RTC clock rate.*
- __STATIC_INLINE void Chip_Clock_SetWDTOSCRate (WDT_OSC_FREQ_T freq, uint32_t div)

  *Set the WDT Oscillator frequency and divider.*
- uint32_t Chip_Clock_GetWDTOSCRate (void)

  *Return estimated watchdog oscillator rate.*
- __STATIC_INLINE uint32_t Chip_Clock_GetFROHFRate (void)

  *Gets the HF-FRO Frequency rate.*
- __STATIC_INLINE void Chip_Clock_SetMain_A_ClockSource (CHIP_SYSCON_MAIN_A_CLKSRC_T src)

  *Set main A system clock source.*
- __STATIC_INLINE void Chip_Clock_SetUSBClockSource (CHIP_SYSCON_USBCLKSRC_T src, uint32_t div)

  *Set USB clock source.*
- __STATIC_INLINE CHIP_SYSCON_USBCLKSRC_T Chip_Clock_GetUSBClockSource (void)

  *Gets the clock source used by USB.*
- __STATIC_INLINE uint32_t Chip_Clock_GetUSBClockDiv (void)

  *Gets the clock divider used by USB.*
- __STATIC_INLINE void Chip_Clock_SetMCLKClockSource (CHIP_SYSCON_MCLKSRC_T src, uint32_t div)

  *Set the MCLK clock source.*
- __STATIC_INLINE uint32_t Chip_Clock_GetMCLKDiv (void)

  *Get MCLK clock div.*
- __STATIC_INLINE CHIP_SYSCON_MCLKSRC_T Chip_Clock_GetMCLKSource (void)

  *Get MCLK clock source.*
- __STATIC_INLINE void Chip_Clock_SetMCLKDirInput (void)

*Set MCLK pin direction to INPUT.*

- __STATIC_INLINE void Chip_Clock_SetMCLKDirOutput (void)

  *Set MCLK pin direction to OUTPUT.*
- __STATIC_INLINE void Chip_Clock_SetMCLKDir (int dir)

  *Set MCLK pin direction to INPUT or OUTPUT.*
- __STATIC_INLINE int Chip_Clock_GetMCLKDir (void)
- __STATIC_INLINE CHIP_SYSCON_MAIN_A_CLKSRC_T Chip_Clock_GetMain_A_ClockSource (void)

  *Returns the main A clock source.*
- uint32_t Chip_Clock_GetMain_A_ClockRate (void)

  *Return main A clock rate.*
- __STATIC_INLINE void Chip_Clock_SetMain_B_ClockSource (CHIP_SYSCON_MAIN_B_CLKSRC_T src)

  *Set main B system clock source.*
- __STATIC_INLINE CHIP_SYSCON_MAIN_B_CLKSRC_T Chip_Clock_GetMain_B_ClockSource (void)

  *Returns the main B clock source.*
- uint32_t Chip_Clock_GetMain_B_ClockRate (void)

  *Return main B clock rate.*
- __STATIC_INLINE void Chip_Clock_SetCLKOUTSource (CHIP_SYSCON_CLKOUTSRC_T src, uint32_t div)

  *Set CLKOUT clock source and divider.*
- __STATIC_INLINE CHIP_SYSCON_CLKOUTSRC_T Chip_Clock_GetCLKOUTSource (void)

  *Get CLKOUT clock source.*
- __STATIC_INLINE uint32_t Chip_Clock_GetCLKOUTDiv (void)

  *Get CLKOUT clock divider.*
- void Chip_Clock_EnablePeriphClock (CHIP_SYSCON_CLOCK_T clk)

  *Enable a system or peripheral clock.*
- void Chip_Clock_DisablePeriphClock (CHIP_SYSCON_CLOCK_T clk)

  *Disable a system or peripheral clock.*
- __STATIC_INLINE void Chip_Clock_SetSysTickClockDiv (uint32_t div)

  *Set system tick clock divider (external CLKIN as SYSTICK reference only)*
- __STATIC_INLINE uint32_t Chip_Clock_GetSysTickClockDiv (void)

  *Returns system tick clock divider.*
- uint32_t Chip_Clock_GetSysTickClockRate (void)

  *Returns the system tick rate as used with the system tick divider.*
- __STATIC_INLINE void Chip_Clock_SetSysClockDiv (uint32_t div)

  *Set system clock divider.*
- __STATIC_INLINE uint32_t Chip_Clock_GetSysClockDiv (void)

  *Get system clock divider.*
- __STATIC_INLINE void Chip_Clock_SetADCClockDiv (uint32_t div)

  *Set system tick clock divider.*
- __STATIC_INLINE uint32_t Chip_Clock_GetADCClockDiv (void)

  *Returns ADC clock divider.*
- __STATIC_INLINE void Chip_Clock_SetFLEXCOMMClockSource (uint32_t idx, CHIP_SYSCON_FLEXCOMMCLKSELSRC_T src)

  *Set the FLEXCOMM clock source.*
- __STATIC_INLINE CHIP_SYSCON_FLEXCOMMCLKSELSRC_T Chip_Clock_GetFLEXCOMMClockSource (uint32_t idx)

  *Returns the FLEXCOMM clock source.*
- uint32_t Chip_Clock_GetFLEXCOMMClockRate (uint32_t id)

  *Return FlexCOMM clock rate.*
- __STATIC_INLINE void Chip_Clock_SetADCClockSource (CHIP_SYSCON_ADCCLKSELSRC_T src)

  *Set the ADC clock source.*

- __STATIC_INLINE CHIP_SYSCON_ADCCLKSELSRC_T Chip_Clock_GetADCClockSource (void)

  *Returns the ADC clock source.*
- uint32_t Chip_Clock_GetADCClockRate (void)

  *Return ADC clock rate.*
- __STATIC_INLINE void Chip_Clock_EnableRTCOsc (void)

  *Enable the RTC 32KHz output.*
- __STATIC_INLINE void Chip_Clock_DisableRTCOsc (void)

  *Disable the RTC 32KHz output.*
- __STATIC_INLINE bool Chip_Clock_GetRTCOsc (void)
- __STATIC_INLINE void Chip_Clock_SetAsyncSysconClockSource (CHIP_ASYNC_SYSCON_SRC_T src)

  *Set asynchronous APB clock source.*
- __STATIC_INLINE CHIP_ASYNC_SYSCON_SRC_T Chip_Clock_GetAsyncSysconClockSource (void)

  *Get asynchronous APB clock source.*
- uint32_t Chip_Clock_GetAsyncSyscon_ClockRate (void)

  *Return asynchronous APB clock rate.*
- __STATIC_INLINE void Chip_Clock_SetMainClockSource (CHIP_SYSCON_MAINCLKSRC_T src)

  *Set main system clock source.*
- CHIP_SYSCON_MAINCLKSRC_T Chip_Clock_GetMainClockSource (void)

  *Get main system clock source.*
- uint32_t Chip_Clock_GetMainClockRate (void)

  *Return main clock rate.*
- uint32_t Chip_Clock_GetSystemClockRate (void)

  *Return system clock rate.*
- uint32_t Chip_Clock_GetFRGInClockRate (void)

  *Get the input clock frequency of FRG.*
- __STATIC_INLINE void Chip_Clock_SetFRGClockSource (CHIP_SYSCON_FRGCLKSRC_T src)

  *Set clock source used by FRG.*
- __STATIC_INLINE CHIP_SYSCON_FRGCLKSRC_T Chip_Clock_GetFRGClockSource (void)

  *Get clock source used by FRG.*
- uint32_t Chip_Clock_GetFRGClockRate (void)

  *Get Fraction Rate Generator (FRG) clock rate.*
- uint32_t Chip_Clock_SetFRGClockRate (uint32_t rate)

  *Set FRG rate to given rate.*

## 6.12.2 Macro Definition Documentation

### 6.12.2.1 #define Chip_Clock_GetIntOscRate( ) SYSCON_FRO12MHZ_FREQ

Definition at line 54 of file clock_5411x.h.

### 6.12.2.2 #define SYSCON_FRO12MHZ_FREQ (12000000)

Definition at line 47 of file clock_5411x.h.

### 6.12.2.3 #define SYSCON_FRO48MHZ_FREQ (48000000)

Definition at line 48 of file clock_5411x.h.

### 6.12.2.4 #define SYSCON_FRO96MHZ_FREQ (96000000)

Definition at line 49 of file clock_5411x.h.

#### 6.12.2.5 #define SYSCON_RTC_FREQ (32768)

Definition at line 51 of file clock_5411x.h.

#### 6.12.2.6 #define SYSCON_WDTOSC_FREQ (500000)

Definition at line 50 of file clock_5411x.h.

### 6.12.3 Enumeration Type Documentation

#### 6.12.3.1 enum CHIP_ASYNC_SYSCON_SRC_T

Clock source selections for the asynchronous APB clock

**Enumerator**

***SYSCON_ASYNC_MAINCLK*** Main System clock

***SYSCON_ASYNC_FRO12MHZ*** 12MHz FRO

Definition at line 621 of file clock_5411x.h.

#### 6.12.3.2 enum CHIP_SYSCON_ADCCLKSELSRC_T

Clock sources for ADC clock source select

**Enumerator**

***SYSCON_ADCCLKSELSRC_MAINCLK*** Main clock

***SYSCON_ADCCLKSELSRC_SYSPLLOUT*** PLL output

***SYSCON_ADCCLKSELSRC_FROHF*** High frequency FRO 48MHz or 96MHz

Definition at line 562 of file clock_5411x.h.

#### 6.12.3.3 enum CHIP_SYSCON_CLKOUTSRC_T

Clock sources for CLKOUT

**Enumerator**

***SYSCON_CLKOUTSRC_MAINCLK*** Main system clock for CLKOUT

***SYSCON_CLKOUTSRC_CLKIN*** CLKIN for CLKOUT

***SYSCON_CLKOUTSRC_WDTOSC*** Watchdog oscillator for CLKOUT

***SYSCON_CLKOUTSRC_FROHF*** 48MHz or 96MHz FRO

***SYSCON_CLKOUTSRC_PLL*** Output of the PLL

***SYSCON_CLKOUTSRC_FRO12MHZ*** 12MHz FRO

***SYSCON_CLKOUTSRC_RTC*** RTC oscillator 32KHz for CLKOUT

***SYSCON_CLKOUTSRC_DISABLED*** Disable clock source for CLKOUT

Definition at line 337 of file clock_5411x.h.

#### 6.12.3.4 enum CHIP_SYSCON_CLOCK_T

System and peripheral clocks enum

**Enumerator**

***SYSCON_CLOCK_ROM*** ROM clock

***SYSCON_CLOCK_SRAM1*** SRAM1 clock

***SYSCON_CLOCK_SRAM2*** SRAM2 clock

***SYSCON_CLOCK_SRAMX*** SRAMX Clock

***SYSCON_CLOCK_FLASH*** FLASH controller clock

***SYSCON_CLOCK_FMC*** FMC clock

***SYSCON_CLOCK_SPIFI*** SPIFI Clock

***SYSCON_CLOCK_INPUTMUX*** Input mux clock

***SYSCON_CLOCK_IOCON*** IOCON clock

***SYSCON_CLOCK_GPIO0*** GPIO0 clock

***SYSCON_CLOCK_GPIO1*** GPIO1 clock

***SYSCON_CLOCK_PINT*** PININT clock

***SYSCON_CLOCK_GINT*** grouped pin interrupt block clock

***SYSCON_CLOCK_DMA*** DMA clock

***SYSCON_CLOCK_CRC*** CRC clock

***SYSCON_CLOCK_WWDT*** WDT clock

***SYSCON_CLOCK_RTC*** RTC clock

***SYSCON_CLOCK_MAILBOX*** Mailbox clock

***SYSCON_CLOCK_ADC0*** ADC0 clock

***SYSCON_CLOCK_MRT*** multi-rate timer clock

***SYSCON_CLOCK_SCT0*** SCT0 clock

***SYSCON_CLOCK_UTICK*** UTICK clock

***SYSCON_CLOCK_FLEXCOMM0*** FLEXCOMM0 Clock

***SYSCON_CLOCK_FLEXCOMM1*** FLEXCOMM1 Clock

***SYSCON_CLOCK_FLEXCOMM2*** FLEXCOMM2 Clock

***SYSCON_CLOCK_FLEXCOMM3*** FLEXCOMM3 Clock

***SYSCON_CLOCK_FLEXCOMM4*** FLEXCOMM4 Clock

***SYSCON_CLOCK_FLEXCOMM5*** FLEXCOMM5 Clock

***SYSCON_CLOCK_FLEXCOMM6*** FLEXCOMM6 Clock

***SYSCON_CLOCK_FLEXCOMM7*** FLEXCOMM7 Clock

***SYSCON_CLOCK_DMIC*** D-MIC Clock

***SYSCON_CLOCK_TIMER2*** TIMER2 clock

***SYSCON_CLOCK_USB*** USB clock

***SYSCON_CLOCK_TIMER0*** TIMER0 clock

***SYSCON_CLOCK_TIMER1*** TIMER1 Clock

***SYSCON_CLOCK_TIMER3*** TIMER3 clock

***SYSCON_CLOCK_TIMER4*** TIMER4 clock

Definition at line 383 of file clock_5411x.h.

#### 6.12.3.5 enum CHIP_SYSCON_FLEXCOMMCLKSELSRC_T

Clock sources for FLEXCOMM clock source select

**Enumerator**

***SYSCON_FLEXCOMMCLKSELSRC_FRO12MHZ*** FRO 12-MHz

***SYSCON_FLEXCOMMCLKSELSRC_FROHF*** HF-FRO 48-MHz or 96-MHz

***SYSCON_FLEXCOMMCLKSELSRC_PLL*** PLL output

***SYSCON_FLEXCOMMCLKSELSRC_MCLK*** MCLK output

***SYSCON_FLEXCOMMCLKSELSRC_FRG*** FRG output

***SYSCON_FLEXCOMMCLKSELSRC_NONE*** NONE output

Definition at line 522 of file clock_5411x.h.

#### 6.12.3.6 enum CHIP_SYSCON_FRGCLKSRC_T

Fractional Divider clock sources.

**Enumerator**

***SYSCON_FRGCLKSRC_MAINCLK*** Main Clock

***SYSCON_FRGCLKSRC_PLL*** Output clock from PLL

***SYSCON_FRGCLKSRC_FRO12MHZ*** FRO 12-MHz

***SYSCON_FRGCLKSRC_FROHF*** FRO High Frequency (48 or 96) MHz

***SYSCON_FRGCLKSRC_NONE*** No clock input

Definition at line 708 of file clock_5411x.h.

#### 6.12.3.7 enum CHIP_SYSCON_MAIN_A_CLKSRC_T

Clock source selections for only the main A system clock. The main A system clock is used as an input into the main B system clock selector. Main clock A only needs to be setup if the main clock A input is used in the main clock system selector.

**Enumerator**

***SYSCON_MAIN_A_CLKSRC_FRO12MHZ*** 12MHz FRO

***SYSCON_MAIN_A_CLKSRCA_CLKIN*** Crystal (main) oscillator in

***SYSCON_MAIN_A_CLKSRCA_WDTOSC*** Watchdog oscillator rate

***SYSCON_MAIN_A_CLKSRCA_FROHF*** 48MHz or 96MHz HF-FRO

Definition at line 136 of file clock_5411x.h.

#### 6.12.3.8 enum CHIP_SYSCON_MAIN_B_CLKSRC_T

Clock sources for only main B system clock

**Enumerator**

***SYSCON_MAIN_B_CLKSRC_MAINCLKSELA*** main clock A

***SYSCON_MAIN_B_CLKSRC_PLL*** System PLL output

***SYSCON_MAIN_B_CLKSRC_RTC*** RTC oscillator 32KHz output

Definition at line 303 of file clock_5411x.h.

#### 6.12.3.9 enum CHIP_SYSCON_MAINCLKSRC_T

Clock sources for main system clock. This is a mix of both main clock A and B selections.

**Enumerator**

***SYSCON_MAINCLKSRC_FRO12MHZ*** 12-MHz FRO

***SYSCON_MAINCLKSRC_CLKIN*** Crystal (main) oscillator in

***SYSCON_MAINCLKSRC_WDTOSC*** Watchdog oscillator rate

***SYSCON_MAINCLKSRC_FROHF*** 48MHz or 96-MHz HF-FRO

***SYSCON_MAINCLKSRC_PLLOUT*** System PLL output

***SYSCON_MAINCLKSRC_RTC*** RTC oscillator 32KHz output

Definition at line 656 of file clock_5411x.h.

#### 6.12.3.10 enum CHIP_SYSCON_MCLKSRC_T

MCLK Clock sources.

**Enumerator**

***SYSCON_MCLKSRC_FROHF*** HF-FRO 48MHz or 96MHz

***SYSCON_MCLKSRC_PLL*** Main pll

***SYSCON_MCLKSRC_MCLKIN*** MCLK INPUT Clock pin set by IOCON

***SYSCON_MCLKSRC_DISABLED*** Disable clock source to MCLK

Definition at line 211 of file clock_5411x.h.

#### 6.12.3.11 enum CHIP_SYSCON_USBCLKSRC_T

USB Clock source.

**Enumerator**

***SYSCON_USBCLKSRC_FROHF*** High frequency FRO 48MHz or 96MHz

***SYSCON_USBCLKSRC_PLL*** USB PLL

***SYSCON_USBCLKSRC_DISABLED*** USB Clock disabled

Definition at line 171 of file clock_5411x.h.

#### 6.12.3.12 enum WDT_OSC_FREQ_T

WDT Osc frequency value table.

**Enumerator**

***WDT_FREQ_RESERVED*** Reserved value

***WDT_FREQ_400000*** WDT Freq 400 KHz

***WDT_FREQ_600000*** WDT Freq 600 KHz

***WDT_FREQ_750000*** WDT Freq 750 KHz

***WDT_FREQ_900000*** WDT Freq 900 KHz

***WDT_FREQ_1000000*** WDT Freq 1.0 MHz

***WDT_FREQ_1200000*** WDT Freq 1.2 MHz

| | |
|---|---|
| ***WDT_FREQ_1300000*** | WDT Freq 1.3 MHz |
| ***WDT_FREQ_1400000*** | WDT Freq 1.4 MHz |
| ***WDT_FREQ_1500000*** | WDT Freq 1.5 MHz |
| ***WDT_FREQ_1600000*** | WDT Freq 1.6 MHz |
| ***WDT_FREQ_1700000*** | WDT Freq 1.7 MHz |
| ***WDT_FREQ_1800000*** | WDT Freq 1.8 MHz |
| ***WDT_FREQ_1900000*** | WDT Freq 1.9 MHz |
| ***WDT_FREQ_2000000*** | WDT Freq 2.0 MHz |
| ***WDT_FREQ_2050000*** | WDT Freq 2.05 MHz |
| ***WDT_FREQ_2100000*** | WDT Freq 2.1 MHz |
| ***WDT_FREQ_2200000*** | WDT Freq 2.2 MHz |
| ***WDT_FREQ_2250000*** | WDT Freq 2.25 MHz |
| ***WDT_FREQ_2300000*** | WDT Freq 2.3 MHz |
| ***WDT_FREQ_2400000*** | WDT Freq 2.4 MHz |
| ***WDT_FREQ_2450000*** | WDT Freq 2.45 MHz |
| ***WDT_FREQ_2500000*** | WDT Freq 2.5 MHz |
| ***WDT_FREQ_2600000*** | WDT Freq 2.6 MHz |
| ***WDT_FREQ_2650000*** | WDT Freq 2.65 MHz |
| ***WDT_FREQ_2700000*** | WDT Freq 2.7 MHz |
| ***WDT_FREQ_2800000*** | WDT Freq 2.8 MHz |
| ***WDT_FREQ_2850000*** | WDT Freq 2.85 MHz |
| ***WDT_FREQ_2900000*** | WDT Freq 2.9 MHz |
| ***WDT_FREQ_2950000*** | WDT Freq 2.95 MHz |
| ***WDT_FREQ_3000000*** | WDT Freq 3.0 MHz |
| ***WDT_FREQ_3050000*** | WDT Freq 3.05 MHz |

Definition at line 75 of file clock_5411x.h.

## 6.12.4 Function Documentation

### 6.12.4.1 void Chip_Clock_DisablePeriphClock ( CHIP_SYSCON_CLOCK_T *clk* )

Disable a system or peripheral clock.

**Parameters**

| | |
|---|---|
| *clk* | : Clock to disable |

**Returns**

Nothing

Definition at line 151 of file clock_5411x.c.

### 6.12.4.2 __STATIC_INLINE void Chip_Clock_DisableRTCOsc ( void )

Disable the RTC 32KHz output.

**Returns**

Nothing

Definition at line 608 of file clock_5411x.h.

#### 6.12.4.3 void Chip_Clock_EnablePeriphClock ( CHIP_SYSCON_CLOCK_T *clk* )

Enable a system or peripheral clock.

**Parameters**

| | |
|---|---|
| *clk* | : Clock to enable |

**Returns**

Nothing

Definition at line 135 of file clock_5411x.c.

#### 6.12.4.4 __STATIC_INLINE void Chip_Clock_EnableRTCOsc ( void )

Enable the RTC 32KHz output.

**Returns**

Nothing

**Note**

This clock can be used for the main clock directly, but do not use this clock with the system PLL.

Definition at line 599 of file clock_5411x.h.

#### 6.12.4.5 __STATIC_INLINE uint32_t Chip_Clock_GetADCClockDiv ( void )

Returns ADC clock divider.

**Returns**

ADC clock divider, 0 = disabled

Definition at line 514 of file clock_5411x.h.

#### 6.12.4.6 uint32_t Chip_Clock_GetADCClockRate ( void )

Return ADC clock rate.

**Returns**

ADC clock rate

Definition at line 185 of file clock_5411x.c.

#### 6.12.4.7 __STATIC_INLINE CHIP_SYSCON_ADCCLKSELSRC_T Chip_Clock_GetADCClockSource ( void )

Returns the ADC clock source.

**Returns**

Returns which clock is used for the ADC clock source

Definition at line 582 of file clock_5411x.h.

#### 6.12.4.8 uint32_t Chip_Clock_GetAsyncSyscon_ClockRate ( void )

Return asynchronous APB clock rate.

**Returns**

Asynchronous APB clock rate

**Note**

Includes adjustments by Async clock divider (ASYNCCLKDIV).

Definition at line 214 of file clock_5411x.c.

#### 6.12.4.9 __STATIC_INLINE CHIP_ASYNC_SYSCON_SRC_T Chip_Clock_GetAsyncSysconClockSource ( void )

Get asynchronous APB clock source.

**Returns**

Clock source for asynchronous APB clock

Definition at line 640 of file clock_5411x.h.

#### 6.12.4.10 __STATIC_INLINE uint32_t Chip_Clock_GetCLKOUTDiv ( void )

Get CLKOUT clock divider.

**Returns**

Clock output source

Definition at line 375 of file clock_5411x.h.

#### 6.12.4.11 __STATIC_INLINE CHIP_SYSCON_CLKOUTSRC_T Chip_Clock_GetCLKOUTSource ( void )

Get CLKOUT clock source.

**Returns**

Clock output source

Definition at line 366 of file clock_5411x.h.

#### 6.12.4.12 __STATIC_INLINE uint32_t Chip_Clock_GetExtClockInRate ( void )

Returns the external clock input rate.

**Returns**

External clock input rate

Definition at line 60 of file clock_5411x.h.

#### 6.12.4.13 uint32_t Chip_Clock_GetFLEXCOMMClockRate ( uint32_t *id* )

Return FlexCOMM clock rate.

**Parameters**

| | |
|---|---|
| *id* | : FlexCOMM ID (Valid range: 0 to 7) |

**Returns**

FlexCOMM clock rate

Definition at line 376 of file clock_5411x.c.

#### 6.12.4.14 __STATIC_INLINE CHIP_SYSCON_FLEXCOMMCLKSELSRC_T Chip_Clock_GetFLEXCOMMClockSource ( uint32_t *idx* )

Returns the FLEXCOMM clock source.

**Parameters**

| | |
|---|---|
| *idx* | : Index of the flexcomm (0 to 7) |

**Returns**

Returns which clock is used for the FLEXCOMM clock source

Definition at line 547 of file clock_5411x.h.

#### 6.12.4.15 uint32_t Chip_Clock_GetFRGClockRate ( void )

Get Fraction Rate Generator (FRG) clock rate.

**Returns**

UART base clock rate

Definition at line 268 of file clock_5411x.c.

#### 6.12.4.16 __STATIC_INLINE CHIP_SYSCON_FRGCLKSRC_T Chip_Clock_GetFRGClockSource ( void )

Get clock source used by FRG.

**Returns**

Clock source used by FRG

**Note**

Definition at line 737 of file clock_5411x.h.

#### 6.12.4.17 uint32_t Chip_Clock_GetFRGInClockRate ( void )

Get the input clock frequency of FRG.

**Returns**

Frequency in Hz on success (0 on failure)

Definition at line 292 of file clock_5411x.c.

#### 6.12.4.18 __STATIC_INLINE uint32_t Chip_Clock_GetFROHFRate ( void )

Gets the HF-FRO Frequency rate.

**Returns**

Nothing

Definition at line 147 of file clock_5411x.h.

#### 6.12.4.19 uint32_t Chip_Clock_GetMain_A_ClockRate ( void )

Return main A clock rate.

**Returns**

main A clock rate in Hz

Definition at line 83 of file clock_5411x.c.

#### 6.12.4.20 __STATIC_INLINE CHIP_SYSCON_MAIN_A_CLKSRC_T Chip_Clock_GetMain_A_ClockSource ( void )

Returns the main A clock source.

**Returns**

Returns which clock is used for the main A

Definition at line 289 of file clock_5411x.h.

#### 6.12.4.21 uint32_t Chip_Clock_GetMain_B_ClockRate ( void )

Return main B clock rate.

**Returns**

main B clock rate

Definition at line 113 of file clock_5411x.c.

#### 6.12.4.22 __STATIC_INLINE CHIP_SYSCON_MAIN_B_CLKSRC_T Chip_Clock_GetMain_B_ClockSource ( void )

Returns the main B clock source.

**Returns**

Returns which clock is used for the main B

Definition at line 323 of file clock_5411x.h.

#### 6.12.4.23 uint32_t Chip_Clock_GetMainClockRate ( void )

Return main clock rate.

**Returns**

main clock rate

Definition at line 244 of file clock_5411x.c.

#### 6.12.4.24 CHIP_SYSCON_MAINCLKSRC_T Chip_Clock_GetMainClockSource ( void )

Get main system clock source.

**Returns**

Clock source for main system

**Note**

Definition at line 224 of file clock_5411x.c.

#### 6.12.4.25 __STATIC_INLINE int Chip_Clock_GetMCLKDir ( void )

Definition at line 280 of file clock_5411x.h.

#### 6.12.4.26 __STATIC_INLINE uint32_t Chip_Clock_GetMCLKDiv ( void )

Get MCLK clock div.

**Returns**

MCLK divider

Definition at line 234 of file clock_5411x.h.

#### 6.12.4.27 __STATIC_INLINE CHIP_SYSCON_MCLKSRC_T Chip_Clock_GetMCLKSource ( void )

Get MCLK clock source.

**Returns**

MCLK clock source

Definition at line 243 of file clock_5411x.h.

#### 6.12.4.28 __STATIC_INLINE bool Chip_Clock_GetRTCOsc ( void )

Definition at line 613 of file clock_5411x.h.

#### 6.12.4.29 __STATIC_INLINE uint32_t Chip_Clock_GetRTCOscRate ( void )

Returns the RTC clock rate.

**Returns**

RTC oscillator clock rate in Hz

Definition at line 69 of file clock_5411x.h.

#### 6.12.4.30 __STATIC_INLINE uint32_t Chip_Clock_GetSysClockDiv ( void )

Get system clock divider.

**Parameters**

| | |
|---|---|
| *None* | |

**Returns**

System clock divider

Definition at line 490 of file clock_5411x.h.

#### 6.12.4.31 uint32_t Chip_Clock_GetSystemClockRate ( void )

Return system clock rate.

**Returns**

system clock rate

**Note**

This is the main clock rate divided by AHBCLKDIV.

Definition at line 261 of file clock_5411x.c.

#### 6.12.4.32 __STATIC_INLINE uint32_t Chip_Clock_GetSysTickClockDiv ( void )

Returns system tick clock divider.

**Returns**

system tick clock divider

Definition at line 462 of file clock_5411x.h.

#### 6.12.4.33 uint32_t Chip_Clock_GetSysTickClockRate ( void )

Returns the system tick rate as used with the system tick divider.

**Returns**

the system tick rate

Definition at line 167 of file clock_5411x.c.

#### 6.12.4.34 __STATIC_INLINE uint32_t Chip_Clock_GetUSBClockDiv ( void )

Gets the clock divider used by USB.

**Returns**

Returns clock divider used by USB

Definition at line 203 of file clock_5411x.h.

#### 6.12.4.35 __STATIC_INLINE CHIP_SYSCON_USBCLKSRC_T Chip_Clock_GetUSBClockSource ( void )

Gets the clock source used by USB.

**Returns**

Returns which clock is used for USB

Definition at line 194 of file clock_5411x.h.

#### 6.12.4.36 uint32_t Chip_Clock_GetWDTOSCRate ( void )

Return estimated watchdog oscillator rate.

**Returns**

Estimated watchdog oscillator rate

**Note**

This rate is accurate to plus or minus 40%.

Definition at line 411 of file clock_5411x.c.

#### 6.12.4.37 __STATIC_INLINE void Chip_Clock_SetADCClockDiv ( uint32_t *div* )

Set system tick clock divider.

**Parameters**

| | |
|---|---|
| *div* | : divider for system clock |

**Returns**

Nothing

**Note**

Use 0 to disable, or a divider value of 1 to 255. The system tick rate is the main system clock divided by this value. Use caution when using the CMSIS SysTick_Config() functions as they typically use SystemCoreClock for setup.

Definition at line 504 of file clock_5411x.h.

#### 6.12.4.38 __STATIC_INLINE void Chip_Clock_SetADCClockSource ( CHIP_SYSCON_ADCCLKSELSRC_T *src* )

Set the ADC clock source.

**Parameters**

| | |
|---|---|
| *src* | : ADC clock source |

**Returns**

Nothing

Definition at line 573 of file clock_5411x.h.

#### 6.12.4.39 __STATIC_INLINE void Chip_Clock_SetAsyncSysconClockSource ( CHIP_ASYNC_SYSCON_SRC_T *src* )

Set asynchronous APB clock source.

**Parameters**

| | |
|---|---|
| *src* | : Clock source for asynchronous APB clock |

**Returns**

Nothing

Definition at line 631 of file clock_5411x.h.

#### 6.12.4.40 __STATIC_INLINE void Chip_Clock_SetCLKOUTSource ( CHIP_SYSCON_CLKOUTSRC_T *src*, uint32_t *div* )

Set CLKOUT clock source and divider.

**Parameters**

| | |
|---|---|
| *src* | : Clock source for CLKOUT (see CHIP_SYSCON_CLKOUTSRC_T) |
| *div* | : divider for CLKOUT clock [Valid range 1 to 256] |

**Returns**

Nothing

**Note**

The CLKOUT clock rate is the clock source divided by the divider.

Definition at line 355 of file clock_5411x.h.

#### 6.12.4.41 __STATIC_INLINE void Chip_Clock_SetFLEXCOMMClockSource ( uint32_t *idx*, CHIP_SYSCON_FLEXCOMM↵CLKSELSRC_T *src* )

Set the FLEXCOMM clock source.

**Parameters**

| | |
|---|---|
| *idx* | : Index of the flexcomm (0 to 7) |
| *src* | : FLEXCOMM clock source (See CHIP_SYSCON_FLEXCOMMCLKSELSRC_T) |

**Returns**

Nothing

Definition at line 537 of file clock_5411x.h.

#### 6.12.4.42 uint32_t Chip_Clock_SetFRGClockRate ( uint32_t *rate* )

Set FRG rate to given rate.

**Returns**

Actual FRG clock rate

**Note**

If FRG is used for UART base clock, *rate* is recommended to be 16 times desired baud rate; **This API must only be called after setting the source using Chip_Clock_SetFRGClockSource()**

Definition at line 322 of file clock_5411x.c.

#### 6.12.4.43 __STATIC_INLINE void Chip_Clock_SetFRGClockSource ( CHIP_SYSCON_FRGCLKSRC_T *src* )

Set clock source used by FRG.

**Returns**

Clock source used by FRG

**Note**

Definition at line 727 of file clock_5411x.h.

#### 6.12.4.44 __STATIC_INLINE void Chip_Clock_SetMain_A_ClockSource ( CHIP_SYSCON_MAIN_A_CLKSRC_T *src* )

Set main A system clock source.

**Parameters**

| | |
|---|---|
| *src* | : Clock source for main A |

**Returns**

Nothing

**Note**

This function only needs to be setup if main clock A will be selected in the Chip_Clock_GetMain_B_ClockRate() function.

Definition at line 163 of file clock_5411x.h.

#### 6.12.4.45 __STATIC_INLINE void Chip_Clock_SetMain_B_ClockSource ( CHIP_SYSCON_MAIN_B_CLKSRC_T *src* )

Set main B system clock source.

**Parameters**

| | |
|---|---|
| *src* | : Clock source for main B |

**Returns**

Nothing

Definition at line 314 of file clock_5411x.h.

#### 6.12.4.46 __STATIC_INLINE void Chip_Clock_SetMainClockSource ( CHIP_SYSCON_MAINCLKSRC_T *src* )

Set main system clock source.

**Parameters**

| | |
|---|---|
| *src* | : Clock source for main system (See CHIP_SYSCON_MAINCLKSRC_T) |

**Returns**

Nothing

Definition at line 670 of file clock_5411x.h.

#### 6.12.4.47 __STATIC_INLINE void Chip_Clock_SetMCLKClockSource ( CHIP_SYSCON_MCLKSRC_T *src*, uint32_t *div* )

Set the MCLK clock source.

**Parameters**

| | |
|---|---|
| *src* | Clock Source for MCLK (see CHIP_SYSCON_MCLKSRC_T) |
| *div* | Value by which the source clock rate be divided (must be greater than 0) |

**Returns**

Nothing

Definition at line 224 of file clock_5411x.h.

#### 6.12.4.48 __STATIC_INLINE void Chip_Clock_SetMCLKDir ( int *dir* )

Set MCLK pin direction to INPUT or OUTPUT.

**Parameters**

| | |
|---|---|
| *dir* | : 0 => INPUT, anything else => OUTPUT |

**Returns**

Nothing

Definition at line 271 of file clock_5411x.h.

#### 6.12.4.49 __STATIC_INLINE void Chip_Clock_SetMCLKDirInput ( void )

Set MCLK pin direction to INPUT.

**Returns**

Nothing

Definition at line 252 of file clock_5411x.h.

#### 6.12.4.50 __STATIC_INLINE void Chip_Clock_SetMCLKDirOutput ( void )

Set MCLK pin direction to OUTPUT.

**Returns**

Nothing

Definition at line 261 of file clock_5411x.h.

#### 6.12.4.51 __STATIC_INLINE void Chip_Clock_SetSysClockDiv ( uint32_t *div* )

Set system clock divider.

**Parameters**

| | |
|---|---|
| *div* | : divider for system clock |

**Returns**

Nothing

**Note**

Use a divider value of 1 to 255. The system clock rate is the main system clock divided by this value.

Definition at line 480 of file clock_5411x.h.

#### 6.12.4.52 __STATIC_INLINE void Chip_Clock_SetSysTickClockDiv ( uint32_t *div* )

Set system tick clock divider (external CLKIN as SYSTICK reference only)

**Parameters**

| | |
|---|---|
| *div* | : divider for system clock |

**Returns**

Nothing

**Note**

Use 0 to disable, or a divider value of 1 to 255. The system tick rate is the external CLKIN rate divided by this value. The extern CLKIN pin signal, divided by the SYSTICKCLKDIV divider, is selected by clearing C↩ LKSOURCE bit 2 in the System Tick CSR register. The core clock must be at least 2.5 times faster than the reference system tick clock otherwise the count values are unpredictable.

Definition at line 453 of file clock_5411x.h.

#### 6.12.4.53 __STATIC_INLINE void Chip_Clock_SetUSBClockSource ( CHIP_SYSCON_USBCLKSRC_T *src*, uint32_t *div* )

Set USB clock source.

**Parameters**

| | |
|---|---|
| *src* | : Clock source for USB (See CHIP_SYSCON_USBCLKSRC_T) |
| *div* | : Value by which the clock must be divided (valid range: 1 - 256) |

**Returns**

Nothing

Definition at line 183 of file clock_5411x.h.

#### 6.12.4.54 __STATIC_INLINE void Chip_Clock_SetWDTOSCRate ( WDT_OSC_FREQ_T *freq*, uint32_t *div* )

Set the WDT Oscillator frequency and divider.

**Parameters**

| | |
|---|---|
| *freq* | : WDT OSC Frequency to set [See WDT_OSC_FREQ_T] |
| *div* | : Divider value [Valid values are 2, 4, 6, 8 ... 64] |

**Returns**

Nothing

**Note**

The actual frequency of the WDT Oscillator can be +/- 40% of frequency set in *freq*.

Definition at line 118 of file clock_5411x.h.

## 6.13 CHIP: LPC5411X Cyclic Redundancy Check Engine driver

### 6.13.1 Detailed Description

#### Data Structures

- struct LPC_CRC_T

  *CRC register block structure.*

#### Macros

- #define CRC_MODE_POLY_BITMASK ((0x03)) /∗∗ CRC polynomial Bit mask ∗/
- #define CRC_MODE_POLY_CCITT (0x00) /∗∗ Select CRC-CCITT polynomial ∗/
- #define CRC_MODE_POLY_CRC16 (0x01) /∗∗ Select CRC-16 polynomial ∗/
- #define CRC_MODE_POLY_CRC32 (0x02) /∗∗ Select CRC-32 polynomial ∗/
- #define CRC_MODE_WRDATA_BITMASK (0x03 << 2) /∗∗ CRC WR_Data Config Bit mask ∗/
- #define CRC_MODE_WRDATA_BIT_RVS (1 << 2) /∗∗ Select Bit order reverse for WR_DATA (per byte) ∗/
- #define CRC_MODE_WRDATA_CMPL (1 << 3) /∗∗ Select One's complement for WR_DATA ∗/
- #define CRC_MODE_SUM_BITMASK (0x03 << 4) /∗∗ CRC Sum Config Bit mask ∗/
- #define CRC_MODE_SUM_BIT_RVS (1 << 4) /∗∗ Select Bit order reverse for CRC_SUM ∗/
- #define CRC_MODE_SUM_CMPL (1 << 5) /∗∗ Select One's complement for CRC_SUM ∗/
- #define MODE_CFG_CCITT (0x00) /∗∗ Pre-defined mode word for default CCITT setup ∗/
- #define MODE_CFG_CRC16 (0x15) /∗∗ Pre-defined mode word for default CRC16 setup ∗/
- #define MODE_CFG_CRC32 (0x36) /∗∗ Pre-defined mode word for default CRC32 setup ∗/
- #define CRC_SEED_CCITT (0x0000FFFF)/∗∗ Initial seed value for CCITT mode ∗/
- #define CRC_SEED_CRC16 (0x00000000)/∗∗ Initial seed value for CRC16 mode ∗/
- #define CRC_SEED_CRC32 (0xFFFFFFFF)/∗∗ Initial seed value for CRC32 mode ∗/

#### Enumerations

- enum CRC_POLY_T { CRC_POLY_CCITT = CRC_MODE_POLY_CCITT, CRC_POLY_CRC16 = CRC_MODE_POLY_CRC16, CRC_POLY_CRC32 = CRC_MODE_POLY_CRC32, CRC_POLY_LAST }

  *CRC polynomial.*

#### Functions

- __STATIC_INLINE void Chip_CRC_Init (LPC_CRC_T ∗pCRC)

  *Initializes the CRC Engine.*
- __STATIC_INLINE void Chip_CRC_Deinit (LPC_CRC_T ∗pCRC)

  *Deinitializes the CRC Engine.*
- __STATIC_INLINE void Chip_CRC_SetPoly (LPC_CRC_T ∗pCRC, CRC_POLY_T poly, uint32_t flags)

  *Set the polynomial used for the CRC calculation.*
- __STATIC_INLINE void Chip_CRC_UseCRC16 (LPC_CRC_T ∗pCRC)

  *Sets up the CRC engine for CRC16 mode.*
- __STATIC_INLINE void Chip_CRC_UseCRC32 (LPC_CRC_T ∗pCRC)

  *Sets up the CRC engine for CRC32 mode.*
- __STATIC_INLINE void Chip_CRC_UseCCITT (LPC_CRC_T ∗pCRC)

  *Sets up the CRC engine for CCITT mode.*
- __STATIC_INLINE void Chip_CRC_UseDefaultConfig (LPC_CRC_T ∗pCRC, CRC_POLY_T poly)

  *Engage the CRC engine with defaults based on the polynomial to be used.*
- __STATIC_INLINE void Chip_CRC_SetMode (LPC_CRC_T ∗pCRC, uint32_t mode)

*Set the CRC Mode bits.*
- __STATIC_INLINE uint32_t Chip_CRC_GetMode (LPC_CRC_T ∗pCRC)

  *Get the CRC Mode bits.*
- __STATIC_INLINE void Chip_CRC_SetSeed (LPC_CRC_T ∗pCRC, uint32_t seed)

  *Set the seed bits used by the CRC_SUM register.*
- __STATIC_INLINE uint32_t Chip_CRC_GetSeed (LPC_CRC_T ∗pCRC)

  *Get the CRC seed value.*
- __STATIC_INLINE void Chip_CRC_Write8 (LPC_CRC_T ∗pCRC, uint8_t data)

  *Convenience function for writing 8-bit data to the CRC engine.*
- __STATIC_INLINE void Chip_CRC_Write16 (LPC_CRC_T ∗pCRC, uint16_t data)

  *Convenience function for writing 16-bit data to the CRC engine.*
- __STATIC_INLINE void Chip_CRC_Write32 (LPC_CRC_T ∗pCRC, uint32_t data)

  *Convenience function for writing 32-bit data to the CRC engine.*
- __STATIC_INLINE uint32_t Chip_CRC_Sum (LPC_CRC_T ∗pCRC)

  *Gets the CRC Sum based on the Mode and Seed as previously configured.*
- __STATIC_INLINE uint32_t Chip_CRC_CRC8 (LPC_CRC_T ∗pCRC, const uint8_t ∗data, uint32_t bytes)

  *Convenience function for computing a standard CCITT checksum from an 8-bit data block.*
- __STATIC_INLINE uint32_t Chip_CRC_CRC16 (LPC_CRC_T ∗pCRC, const uint16_t ∗data, uint32_t hwords)

  *Convenience function for computing a standard CRC16 checksum from 16-bit data block.*
- __STATIC_INLINE uint32_t Chip_CRC_CRC32 (LPC_CRC_T ∗pCRC, const uint32_t ∗data, uint32_t words)

  *Convenience function for computing a standard CRC32 checksum from 32-bit data block.*

## 6.13.2 Macro Definition Documentation

### 6.13.2.1 #define CRC_MODE_POLY_BITMASK ((0x03)) /∗∗ CRC polynomial Bit mask ∗/

Definition at line 62 of file crc_5411x.h.

### 6.13.2.2 #define CRC_MODE_POLY_CCITT (0x00) /∗∗ Select CRC-CCITT polynomial ∗/

Definition at line 63 of file crc_5411x.h.

### 6.13.2.3 #define CRC_MODE_POLY_CRC16 (0x01) /∗∗ Select CRC-16 polynomial ∗/

Definition at line 64 of file crc_5411x.h.

### 6.13.2.4 #define CRC_MODE_POLY_CRC32 (0x02) /∗∗ Select CRC-32 polynomial ∗/

Definition at line 65 of file crc_5411x.h.

### 6.13.2.5 #define CRC_MODE_SUM_BIT_RVS (1 << 4) /∗∗ Select Bit order reverse for CRC_SUM ∗/

Definition at line 70 of file crc_5411x.h.

### 6.13.2.6 #define CRC_MODE_SUM_BITMASK (0x03 << 4) /∗∗ CRC Sum Config Bit mask ∗/

Definition at line 69 of file crc_5411x.h.

#### 6.13.2.7 #define CRC_MODE_SUM_CMPL (1 $<<$ 5) /$**$ Select One's complement for CRC_SUM $*$/

Definition at line 71 of file crc_5411x.h.

#### 6.13.2.8 #define CRC_MODE_WRDATA_BIT_RVS (1 $<<$ 2) /$**$ Select Bit order reverse for WR_DATA (per byte) $*$/

Definition at line 67 of file crc_5411x.h.

#### 6.13.2.9 #define CRC_MODE_WRDATA_BITMASK (0x03 $<<$ 2) /$**$ CRC WR_Data Config Bit mask $*$/

Definition at line 66 of file crc_5411x.h.

#### 6.13.2.10 #define CRC_MODE_WRDATA_CMPL (1 $<<$ 3) /$**$ Select One's complement for WR_DATA $*$/

Definition at line 68 of file crc_5411x.h.

#### 6.13.2.11 #define CRC_SEED_CCITT (0x0000FFFF)/$**$ Initial seed value for CCITT mode $*$/

Definition at line 77 of file crc_5411x.h.

#### 6.13.2.12 #define CRC_SEED_CRC16 (0x00000000)/$**$ Initial seed value for CRC16 mode $*$/

Definition at line 78 of file crc_5411x.h.

#### 6.13.2.13 #define CRC_SEED_CRC32 (0xFFFFFFFF)/$**$ Initial seed value for CRC32 mode $*$/

Definition at line 79 of file crc_5411x.h.

#### 6.13.2.14 #define MODE_CFG_CCITT (0x00) /$**$ Pre-defined mode word for default CCITT setup $*$/

Definition at line 73 of file crc_5411x.h.

#### 6.13.2.15 #define MODE_CFG_CRC16 (0x15) /$**$ Pre-defined mode word for default CRC16 setup $*$/

Definition at line 74 of file crc_5411x.h.

#### 6.13.2.16 #define MODE_CFG_CRC32 (0x36) /$**$ Pre-defined mode word for default CRC32 setup $*$/

Definition at line 75 of file crc_5411x.h.

### 6.13.3 Enumeration Type Documentation

#### 6.13.3.1 enum CRC_POLY_T

CRC polynomial.

**Enumerator**

***CRC_POLY_CCITT*** CRC-CCIT polynomial

***CRC_POLY_CRC16*** CRC-16 polynomial

***CRC_POLY_CRC32*** CRC-32 polynomial

***CRC_POLY_LAST***

Definition at line 84 of file crc_5411x.h.

## 6.13.4 Function Documentation

### 6.13.4.1 __STATIC_INLINE uint32_t Chip_CRC_CRC16 ( LPC_CRC_T ∗ *pCRC,* const uint16_t ∗ *data,* uint32_t *hwords* )

Convenience function for computing a standard CRC16 checksum from 16-bit data block.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |
| *data* | : Pointer to the block of 16-bit data |
| *hwords* | : The number of 16 byte entries pointed to by data |

**Returns**

Check sum value

Definition at line 308 of file crc_5411x.h.

### 6.13.4.2 __STATIC_INLINE uint32_t Chip_CRC_CRC32 ( LPC_CRC_T ∗ *pCRC,* const uint32_t ∗ *data,* uint32_t *words* )

Convenience function for computing a standard CRC32 checksum from 32-bit data block.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |
| *data* | : Pointer to the block of 32-bit data |
| *words* | : The number of 32-bit entries pointed to by data |

**Returns**

Check sum value

Definition at line 327 of file crc_5411x.h.

### 6.13.4.3 __STATIC_INLINE uint32_t Chip_CRC_CRC8 ( LPC_CRC_T ∗ *pCRC,* const uint8_t ∗ *data,* uint32_t *bytes* )

Convenience function for computing a standard CCITT checksum from an 8-bit data block.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |
| *data* | : Pointer to the block of 8-bit data |
| *bytes* | : The number of bytes pointed to by data |

**Returns**

Check sum value

Definition at line 289 of file crc_5411x.h.

### 6.13.4.4 __STATIC_INLINE void Chip_CRC_Deinit ( LPC_CRC_T ∗ *pCRC* )

Deinitializes the CRC Engine.

**Parameters**

|  |  |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

Nothing

Definition at line 114 of file crc_5411x.h.

#### 6.13.4.5 __STATIC_INLINE uint32_t Chip_CRC_GetMode ( LPC_CRC_T ∗ *pCRC* )

Get the CRC Mode bits.

**Parameters**

|  |  |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

The current value of the CRC Mode bits

Definition at line 213 of file crc_5411x.h.

#### 6.13.4.6 __STATIC_INLINE uint32_t Chip_CRC_GetSeed ( LPC_CRC_T ∗ *pCRC* )

Get the CRC seed value.

**Parameters**

|  |  |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

Seed value

Definition at line 234 of file crc_5411x.h.

#### 6.13.4.7 __STATIC_INLINE void Chip_CRC_Init ( LPC_CRC_T ∗ *pCRC* )

Initializes the CRC Engine.

**Parameters**

|  |  |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

Nothing

Definition at line 96 of file crc_5411x.h.

#### 6.13.4.8 __STATIC_INLINE void Chip_CRC_SetMode ( LPC_CRC_T ∗ *pCRC,* uint32_t *mode* )

Set the CRC Mode bits.

**Parameters**

| | |
|---:|---|
| *pCRC* | : Pointer to the crc register block |
| *mode* | : Mode value |

**Returns**

Nothing

Definition at line 203 of file crc_5411x.h.

#### 6.13.4.9 __STATIC_INLINE void Chip_CRC_SetPoly ( LPC_CRC_T ∗ *pCRC*, CRC_POLY_T *poly*, uint32_t *flags* )

Set the polynomial used for the CRC calculation.

**Parameters**

| | |
|---:|---|
| *pCRC* | : Pointer to the crc register block |
| *poly* | : The enumerated polynomial to be used |
| *flags* | : An Or'ed value of flags that setup the mode |

**Returns**

Nothing

**Note**

Flags for setting up the mode word include CRC_MODE_WRDATA_BIT_RVS, CRC_MODE_WRDATA_C↩ MPL, CRC_MODE_SUM_BIT_RVS, and CRC_MODE_SUM_CMPL.

Definition at line 135 of file crc_5411x.h.

#### 6.13.4.10 __STATIC_INLINE void Chip_CRC_SetSeed ( LPC_CRC_T ∗ *pCRC*, uint32_t *seed* )

Set the seed bits used by the CRC_SUM register.

**Parameters**

| | |
|---:|---|
| *pCRC* | : Pointer to the crc register block |
| *seed* | : Seed value |

**Returns**

Nothing

Definition at line 224 of file crc_5411x.h.

#### 6.13.4.11 __STATIC_INLINE uint32_t Chip_CRC_Sum ( LPC_CRC_T ∗ *pCRC* )

Gets the CRC Sum based on the Mode and Seed as previously configured.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

CRC Checksum value

Definition at line 277 of file crc_5411x.h.

#### 6.13.4.12 __STATIC_INLINE void Chip_CRC_UseCCITT ( LPC_CRC_T ∗ *pCRC* )

Sets up the CRC engine for CCITT mode.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

Nothing

Definition at line 167 of file crc_5411x.h.

#### 6.13.4.13 __STATIC_INLINE void Chip_CRC_UseCRC16 ( LPC_CRC_T ∗ *pCRC* )

Sets up the CRC engine for CRC16 mode.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

Nothing

Definition at line 145 of file crc_5411x.h.

#### 6.13.4.14 __STATIC_INLINE void Chip_CRC_UseCRC32 ( LPC_CRC_T ∗ *pCRC* )

Sets up the CRC engine for CRC32 mode.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |

**Returns**

Nothing

Definition at line 156 of file crc_5411x.h.

#### 6.13.4.15 __STATIC_INLINE void Chip_CRC_UseDefaultConfig ( LPC_CRC_T ∗ *pCRC,* CRC_POLY_T *poly* )

Engage the CRC engine with defaults based on the polynomial to be used.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |
| *poly* | : The enumerated polynomial to be used |

**Returns**

Nothing

Definition at line 179 of file crc_5411x.h.

#### 6.13.4.16 __STATIC_INLINE void Chip_CRC_Write16 ( LPC_CRC_T ∗ *pCRC*, uint16_t *data* )

Convenience function for writing 16-bit data to the CRC engine.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |
| *data* | : 16-bit data to write |

**Returns**

Nothing

Definition at line 256 of file crc_5411x.h.

#### 6.13.4.17 __STATIC_INLINE void Chip_CRC_Write32 ( LPC_CRC_T ∗ *pCRC*, uint32_t *data* )

Convenience function for writing 32-bit data to the CRC engine.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |
| *data* | : 32-bit data to write |

**Returns**

Nothing

Definition at line 267 of file crc_5411x.h.

#### 6.13.4.18 __STATIC_INLINE void Chip_CRC_Write8 ( LPC_CRC_T ∗ *pCRC*, uint8_t *data* )

Convenience function for writing 8-bit data to the CRC engine.

**Parameters**

| | |
|---|---|
| *pCRC* | : Pointer to the crc register block |
| *data* | : 8-bit data to write |

**Returns**

Nothing

Definition at line 245 of file crc_5411x.h.

# 6.14 CHIP: LPC5411X DMA Controller driver channel specific functions (legacy)

## 6.14.1 Detailed Description

### Data Structures

- struct DMA_CHDESC_T

### Functions

- __STATIC_INLINE void Chip_DMA_SetupChannelConfig (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32↩_t cfg)

  *Setup a DMA channel configuration.*
- __STATIC_INLINE uint32_t Chip_DMA_GetChannelStatus (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Returns channel specific status flags.*
- __STATIC_INLINE void Chip_DMA_SetupChannelTransfer (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t cfg)

  *Setup a DMA channel transfer configuration.*
- __STATIC_INLINE void Chip_DMA_SetTranBits (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t mask)

  *Set DMA transfer register interrupt bits (safe)*
- __STATIC_INLINE void Chip_DMA_ClearTranBits (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t mask)

  *Clear DMA transfer register interrupt bits (safe)*
- __STATIC_INLINE void Chip_DMA_SetupChannelTransferSize (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t trans)

  *Update the transfer size in an existing DMA channel transfer configuration.*
- __STATIC_INLINE void Chip_DMA_SetChannelValid (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets a DMA channel configuration as valid.*
- __STATIC_INLINE void Chip_DMA_SetChannelInValid (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets a DMA channel configuration as invalid.*
- __STATIC_INLINE void Chip_DMA_SWTriggerChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Performs a software trigger of the DMA channel.*
- __STATIC_INLINE bool Chip_DMA_SetupTranChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, DMA_C↩HDESC_T ∗desc)

  *Sets up a DMA channel with the passed DMA transfer descriptor.*

### Variables

- DMA_CHDESC_T Chip_DMA_Table [MAX_DMA_CHANNEL]

## 6.14.2 Function Documentation

### 6.14.2.1 __STATIC_INLINE void Chip_DMA_ClearTranBits ( LPC_DMA_T ∗ *pDMA*, DMA_CHID_T *ch*, uint32_t *mask* )

Clear DMA transfer register interrupt bits (safe)

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

| | |
|---:|---|
| *ch* | : DMA channel ID |
| *mask* | : Bits to clear |

**Returns**

Nothing

**Note**

This function safely clears bits in the DMA channel specific XFERCFG register.

Definition at line 624 of file dma_5411x.h.

#### 6.14.2.2 __STATIC_INLINE uint32_t Chip_DMA_GetChannelStatus ( LPC_DMA_T * *pDMA*, DMA_CHID_T *ch* )

Returns channel specific status flags.

**Parameters**

| | |
|---:|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

AN Or'ed value of DMA_CTLSTAT_VALIDPENDING and DMA_CTLSTAT_TRIG

Definition at line 574 of file dma_5411x.h.

#### 6.14.2.3 __STATIC_INLINE void Chip_DMA_SetChannelInValid ( LPC_DMA_T * *pDMA*, DMA_CHID_T *ch* )

Sets a DMA channel configuration as invalid.

**Parameters**

| | |
|---:|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 659 of file dma_5411x.h.

#### 6.14.2.4 __STATIC_INLINE void Chip_DMA_SetChannelValid ( LPC_DMA_T * *pDMA*, DMA_CHID_T *ch* )

Sets a DMA channel configuration as valid.

**Parameters**

| | |
|---:|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 648 of file dma_5411x.h.

#### 6.14.2.5 __STATIC_INLINE void Chip_DMA_SetTranBits ( LPC_DMA_T ∗ *pDMA*, DMA_CHID_T *ch*, uint32_t *mask* )

Set DMA transfer register interrupt bits (safe)

**Parameters**

| | |
|---:|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |
| *mask* | : Bits to set |

**Returns**

Nothing

**Note**

This function safely sets bits in the DMA channel specific XFERCFG register.

Definition at line 610 of file dma_5411x.h.

#### 6.14.2.6 __STATIC_INLINE void Chip_DMA_SetupChannelConfig ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch,* uint32_t *cfg* )

Setup a DMA channel configuration.

**Parameters**

| | |
|---:|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |
| *cfg* | : An Or'ed value of DMA_CFG_∗ values that define the channel's configuration |

**Returns**

Nothing

**Note**

This function sets up all configurable options for the DMA channel. These options are usually set once for a channel and then unchanged.

The following example show how to configure the channel for peripheral DMA requests, burst transfer size of 1 (in 'transfers', not bytes), continuous reading of the same source address, incrementing destination address, and highest channel priority.
Example: Chip_DMA_SetupChannelConfig(pDMA, SSP0_RX_DMA, (DMA_CFG_PERIPHREQEN | DMA_CFG_TRIGBURST_BURST | DMA_CFG_BURSTPOWER_1 | DMA_CFG_SRCBURSTWRAP | DMA_CFG_CHPRIORITY(0)));

The following example show how to configure the channel for an external trigger from the imput mux with low edge polarity, a burst transfer size of 8, incrementing source and destination addresses, and lowest channel priority.
Example: Chip_DMA_SetupChannelConfig(pDMA, DMA_CH14, (DMA_CFG_HWTRIGEN | DMA_CFG_TRIGPOL_LOW | DMA_CFG_TRIGTYPE_EDGE | DMA_CFG_TRIGBURST_BURST | DMA_CFG_BURSTPOWER_8 | DMA_CFG_CHPRIORITY(3)));

For non-peripheral DMA triggering (DMA_CFG_HWTRIGEN definition), use the DMA input mux functions to configure the DMA trigger source for a DMA channel.

Definition at line 563 of file dma_5411x.h.

#### 6.14.2.7 __STATIC_INLINE void Chip_DMA_SetupChannelTransfer ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch,* uint32_t *cfg* )

Setup a DMA channel transfer configuration.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |
| *cfg* | : An Or'ed value of DMA_XFERCFG_∗ values that define the channel's transfer configuration |

**Returns**

Nothing

**Note**

This function sets up the transfer configuration for the DMA channel.

The following example show how to configure the channel's transfer for multiple transfer descriptors (ie, ping-pong), interrupt 'A' trigger on transfer descriptor completion, 128 byte size transfers, and source and destination address increment.
Example: Chip_DMA_SetupChannelTransfer(pDMA, SSP0_RX_DMA, (DMA_XFERCFG_CFGVALID | DMA_X↩ FERCFG_RELOAD | DMA_XFERCFG_SETINTA | DMA_XFERCFG_WIDTH_8 | DMA_XFERCFG_SRCINC_1 | DMA_XFERCFG_DSTINC_1 | DMA_XFERCFG_XFERCOUNT(128)));

Definition at line 596 of file dma_5411x.h.

#### 6.14.2.8 __STATIC_INLINE void Chip_DMA_SetupChannelTransferSize ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch,* uint32_t *trans* )

Update the transfer size in an existing DMA channel transfer configuration.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |
| *trans* | : Number of transfers to update the transfer configuration to (1 - 1023) |

**Returns**

Nothing

Definition at line 636 of file dma_5411x.h.

#### 6.14.2.9 __STATIC_INLINE bool Chip_DMA_SetupTranChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch,* DMA_CHDESC_T ∗ *desc* )

Sets up a DMA channel with the passed DMA transfer descriptor.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |
| *desc* | : Pointer to DMA transfer descriptor |

**Returns**

false if the DMA channel was active, otherwise true

**Note**

This function will set the DMA descriptor in the SRAM table to the the passed descriptor. This function is only meant to be used when the DMA channel is not active and can be used to setup the initial transfer for a linked list or ping-pong buffer or just a single transfer without a next descriptor.

If using this function to write the initial transfer descriptor in a linked list or ping-pong buffer configuration, it should contain a non-NULL 'next' field pointer.

Definition at line 691 of file dma_5411x.h.

#### 6.14.2.10 __STATIC_INLINE void Chip_DMA_SWTriggerChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch* )

Performs a software trigger of the DMA channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 670 of file dma_5411x.h.

### 6.14.3 Variable Documentation

#### 6.14.3.1 DMA_CHDESC_T Chip_DMA_Table[MAX_DMA_CHANNEL]

## 6.15 CHIP: LPC5411X DMA Controller driver common channel functions (legacy)

### 6.15.1 Detailed Description

### Functions

- __STATIC_INLINE void Chip_DMA_EnableChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Enables a single DMA channel.*
- __STATIC_INLINE void Chip_DMA_DisableChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Disables a single DMA channel.*
- __STATIC_INLINE uint32_t Chip_DMA_GetEnabledChannels (LPC_DMA_T ∗pDMA)

  *Returns all enabled DMA channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetActiveChannels (LPC_DMA_T ∗pDMA)

  *Returns all active DMA channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetBusyChannels (LPC_DMA_T ∗pDMA)

  *Returns all busy DMA channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetErrorIntChannels (LPC_DMA_T ∗pDMA)

  *Returns pending error interrupt status for all DMA channels.*
- __STATIC_INLINE void Chip_DMA_ClearErrorIntChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Clears a pending error interrupt status for a single DMA channel.*
- __STATIC_INLINE void Chip_DMA_EnableIntChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Enables a single DMA channel's interrupt used in common DMA interrupt.*
- __STATIC_INLINE void Chip_DMA_DisableIntChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Disables a single DMA channel's interrupt used in common DMA interrupt.*
- __STATIC_INLINE uint32_t Chip_DMA_GetEnableIntChannels (LPC_DMA_T ∗pDMA)

  *Returns all enabled interrupt channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetActiveIntAChannels (LPC_DMA_T ∗pDMA)

  *Returns active A interrupt status for all channels.*
- __STATIC_INLINE void Chip_DMA_ClearActiveIntAChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Clears active A interrupt status for a single channel.*
- __STATIC_INLINE uint32_t Chip_DMA_GetActiveIntBChannels (LPC_DMA_T ∗pDMA)

  *Returns active B interrupt status for all channels.*
- __STATIC_INLINE void Chip_DMA_ClearActiveIntBChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Clears active B interrupt status for a single channel.*
- __STATIC_INLINE void Chip_DMA_SetValidChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets the VALIDPENDING control bit for a single channel.*
- __STATIC_INLINE void Chip_DMA_SetTrigChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets the TRIG bit for a single channel.*
- __STATIC_INLINE void Chip_DMA_AbortChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Aborts a DMA operation for a single channel.*

### 6.15.2 Function Documentation

#### 6.15.2.1 __STATIC_INLINE void Chip_DMA_AbortChannel ( **LPC_DMA_T** ∗ *pDMA,* **DMA_CHID_T** *ch* )

Aborts a DMA operation for a single channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

**Note**

To abort a channel, the channel should first be disabled. Then wait until the channel is no longer busy by checking the corresponding bit in BUSY. Finally, abort the channel operation. This prevents the channel from restarting an incomplete operation when it is enabled again.

Definition at line 509 of file dma_5411x.h.

#### 6.15.2.2 __STATIC_INLINE void Chip_DMA_ClearActiveIntAChannel ( LPC_DMA_T ∗ *pDMA*, DMA_CHID_T *ch* )

Clears active A interrupt status for a single channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 443 of file dma_5411x.h.

#### 6.15.2.3 __STATIC_INLINE void Chip_DMA_ClearActiveIntBChannel ( LPC_DMA_T ∗ *pDMA*, DMA_CHID_T *ch* )

Clears active B interrupt status for a single channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 468 of file dma_5411x.h.

#### 6.15.2.4 __STATIC_INLINE void Chip_DMA_ClearErrorIntChannel ( LPC_DMA_T ∗ *pDMA*, DMA_CHID_T *ch* )

Clears a pending error interrupt status for a single DMA channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 381 of file dma_5411x.h.

#### 6.15.2.5 __STATIC_INLINE void Chip_DMA_DisableChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch* )

Disables a single DMA channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 312 of file dma_5411x.h.

#### 6.15.2.6 __STATIC_INLINE void Chip_DMA_DisableIntChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch* )

Disables a single DMA channel's interrupt used in common DMA interrupt.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 403 of file dma_5411x.h.

#### 6.15.2.7 __STATIC_INLINE void Chip_DMA_EnableChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch* )

Enables a single DMA channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 301 of file dma_5411x.h.

#### 6.15.2.8 __STATIC_INLINE void Chip_DMA_EnableIntChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch* )

Enables a single DMA channel's interrupt used in common DMA interrupt.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

Definition at line 392 of file dma_5411x.h.

### 6.15.2.9 __STATIC_INLINE uint32_t Chip_DMA_GetActiveChannels ( LPC_DMA_T ∗ *pDMA* )

Returns all active DMA channels.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

An Or'ed value of all active DMA channels (0 - 15)

**Note**

A high values in bits 0 .. 15 in the return values indicates that the channel for that bit (bit 0 = channel 0, bit 1 - channel 1, etc.) is active. A low state is inactive. A active channel indicates that a DMA operation has been started but not yet fully completed.

Definition at line 340 of file dma_5411x.h.

### 6.15.2.10 __STATIC_INLINE uint32_t Chip_DMA_GetActiveIntAChannels ( LPC_DMA_T ∗ *pDMA* )

Returns active A interrupt status for all channels.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

Nothing

**Note**

A high values in bits 0 .. 15 in the return values indicates that the channel for that bit (bit 0 = channel 0, bit 1 - channel 1, etc.) has an active A interrupt for the channel. A low state indicates that the A interrupt is not active.

Definition at line 432 of file dma_5411x.h.

### 6.15.2.11 __STATIC_INLINE uint32_t Chip_DMA_GetActiveIntBChannels ( LPC_DMA_T ∗ *pDMA* )

Returns active B interrupt status for all channels.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

Nothing

**Note**

A high values in bits 0 .. 15 in the return values indicates that the channel for that bit (bit 0 = channel 0, bit 1 - channel 1, etc.) has an active B interrupt for the channel. A low state indicates that the B interrupt is not active.

Definition at line 457 of file dma_5411x.h.

#### 6.15.2.12 __STATIC_INLINE uint32_t Chip_DMA_GetBusyChannels ( LPC_DMA_T ∗ *pDMA* )

Returns all busy DMA channels.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

An Or'ed value of all busy DMA channels (0 - 15)

**Note**

A high values in bits 0 .. 15 in the return values indicates that the channel for that bit (bit 0 = channel 0, bit 1 - channel 1, etc.) is busy. A low state is not busy. A DMA channel is considered busy when there is any operation related to that channel in the DMA controllers internal pipeline.

Definition at line 356 of file dma_5411x.h.

#### 6.15.2.13 __STATIC_INLINE uint32_t Chip_DMA_GetEnabledChannels ( LPC_DMA_T ∗ *pDMA* )

Returns all enabled DMA channels.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

An Or'ed value of all enabled DMA channels (0 - 15)

**Note**

A high values in bits 0 .. 15 in the return values indicates that the channel for that bit (bit 0 = channel 0, bit 1 - channel 1, etc.) is enabled. A low state is disabled.

Definition at line 325 of file dma_5411x.h.

#### 6.15.2.14 __STATIC_INLINE uint32_t Chip_DMA_GetEnableIntChannels ( LPC_DMA_T ∗ *pDMA* )

Returns all enabled interrupt channels.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

Nothing

**Note**

A high values in bits 0 .. 15 in the return values indicates that the channel for that bit (bit 0 = channel 0, bit 1 - channel 1, etc.) has an enabled interrupt for the channel. A low state indicates that the DMA channel will not contribute to the common DMA interrupt status.

Definition at line 418 of file dma_5411x.h.

#### 6.15.2.15 __STATIC_INLINE uint32_t Chip_DMA_GetErrorIntChannels ( LPC_DMA_T ∗ *pDMA* )

Returns pending error interrupt status for all DMA channels.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

An Or'ed value of all channels (0 - 15) error interrupt status

**Note**

A high values in bits 0 .. 15 in the return values indicates that the channel for that bit (bit 0 = channel 0, bit 1 - channel 1, etc.) has a pending error interrupt. A low state indicates no error interrupt.

Definition at line 370 of file dma_5411x.h.

#### 6.15.2.16 __STATIC_INLINE void Chip_DMA_SetTrigChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch* )

Sets the TRIG bit for a single channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

**Note**

See the User Manual for more information for what this bit does.

Definition at line 493 of file dma_5411x.h.

#### 6.15.2.17 __STATIC_INLINE void Chip_DMA_SetValidChannel ( LPC_DMA_T ∗ *pDMA,* DMA_CHID_T *ch* )

Sets the VALIDPENDING control bit for a single channel.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *ch* | : DMA channel ID |

**Returns**

Nothing

**Note**

See the User Manual for more information for what this bit does.

Definition at line 481 of file dma_5411x.h.

## 6.16 CHIP: LPC5411X DMA Controller driver common functions (legacy)

### 6.16.1 Detailed Description

### Functions

- __STATIC_INLINE void Chip_DMA_Init (LPC_DMA_T ∗pDMA)

  *Initialize DMA controller.*
- __STATIC_INLINE void Chip_DMA_DeInit (LPC_DMA_T ∗pDMA)

  *De-Initialize DMA controller.*
- __STATIC_INLINE void Chip_DMA_Enable (LPC_DMA_T ∗pDMA)

  *Enable DMA controller.*
- __STATIC_INLINE void Chip_DMA_Disable (LPC_DMA_T ∗pDMA)

  *Disable DMA controller.*
- __STATIC_INLINE uint32_t Chip_DMA_GetIntStatus (LPC_DMA_T ∗pDMA)

  *Get pending interrupt or error interrupts.*
- __STATIC_INLINE void Chip_DMA_SetSRAMBase (LPC_DMA_T ∗pDMA, uint32_t base)

  *Set DMA controller SRAM base address.*
- __STATIC_INLINE uint32_t Chip_DMA_GetSRAMBase (LPC_DMA_T ∗pDMA)

  *Returns DMA controller SRAM base address.*

### 6.16.2 Function Documentation

#### 6.16.2.1 __STATIC_INLINE void Chip_DMA_DeInit ( LPC_DMA_T ∗ *pDMA* )

De-Initialize DMA controller.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

Nothing

Definition at line 221 of file dma_5411x.h.

#### 6.16.2.2 __STATIC_INLINE void Chip_DMA_Disable ( LPC_DMA_T ∗ *pDMA* )

Disable DMA controller.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

Nothing

Definition at line 242 of file dma_5411x.h.

#### 6.16.2.3 __STATIC_INLINE void Chip_DMA_Enable ( LPC_DMA_T ∗ *pDMA* )

Enable DMA controller.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

Nothing

Definition at line 232 of file dma_5411x.h.

#### 6.16.2.4 __STATIC_INLINE uint32_t Chip_DMA_GetIntStatus ( LPC_DMA_T ∗ *pDMA* )

Get pending interrupt or error interrupts.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

An Or'ed value of DMA_INTSTAT_∗ types

**Note**

If any DMA channels have an active interrupt or error interrupt pending, this functional will a common status that applies to all channels.

Definition at line 255 of file dma_5411x.h.

#### 6.16.2.5 __STATIC_INLINE uint32_t Chip_DMA_GetSRAMBase ( LPC_DMA_T ∗ *pDMA* )

Returns DMA controller SRAM base address.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

The base address where the DMA descriptors are stored

Definition at line 282 of file dma_5411x.h.

#### 6.16.2.6 __STATIC_INLINE void Chip_DMA_Init ( LPC_DMA_T ∗ *pDMA* )

Initialize DMA controller.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |

**Returns**

Nothing

Definition at line 210 of file dma_5411x.h.

#### 6.16.2.7 __STATIC_INLINE void Chip_DMA_SetSRAMBase ( LPC_DMA_T ∗ *pDMA,* uint32_t *base* )

Set DMA controller SRAM base address.

**Parameters**

| | |
|---|---|
| *pDMA* | : The base of DMA controller on the chip |
| *base* | : The base address where the DMA descriptors will be stored |

**Returns**

Nothing

**Note**

A 256 byte block of memory aligned on a 256 byte boundary must be provided for this function. It sets the base address used for DMA descriptor table (16 descriptors total that use 16 bytes each).

A pre-defined table with correct alignment can be used for this function by calling Chip_DMA_SetSRAMBase(LP↩
C_DMA, DMA_ADDR(Chip_DMA_Table));

Definition at line 272 of file dma_5411x.h.

## 6.17 CHIP: LPC5411X DMA Engine driver (legacy)

### 6.17.1 Detailed Description

#### Data Structures

- struct LPC_DMA_COMMON_T

  *DMA Controller shared registers structure.*
- struct LPC_DMA_CHANNEL_T

  *DMA Controller shared registers structure.*
- struct LPC_DMA_T

  *DMA Controller register block structure.*

#### Modules

- CHIP: LPC5411X DMA Controller driver channel specific functions (legacy)
- CHIP: LPC5411X DMA Controller driver common channel functions (legacy)
- CHIP: LPC5411X DMA Controller driver common functions (legacy)

#### Macros

- #define MAX_DMA_CHANNEL (20)
- #define DMA_INTSTAT_ACTIVEINT 0x2
- #define DMA_INTSTAT_ACTIVEERRINT 0x4
- #define DMA_ADDR(addr) ((uint32_t) (addr))
- #define DMA_CFG_PERIPHREQEN (1 << 0)
- #define DMA_CFG_HWTRIGEN (1 << 1)
- #define DMA_CFG_TRIGPOL_LOW (0 << 4)
- #define DMA_CFG_TRIGPOL_HIGH (1 << 4)
- #define DMA_CFG_TRIGTYPE_EDGE (0 << 5)
- #define DMA_CFG_TRIGTYPE_LEVEL (1 << 5)
- #define DMA_CFG_TRIGBURST_SNGL (0 << 6)
- #define DMA_CFG_TRIGBURST_BURST (1 << 6)
- #define DMA_CFG_BURSTPOWER_1 (0 << 8)
- #define DMA_CFG_BURSTPOWER_2 (1 << 8)
- #define DMA_CFG_BURSTPOWER_4 (2 << 8)
- #define DMA_CFG_BURSTPOWER_8 (3 << 8)
- #define DMA_CFG_BURSTPOWER_16 (4 << 8)
- #define DMA_CFG_BURSTPOWER_32 (5 << 8)
- #define DMA_CFG_BURSTPOWER_64 (6 << 8)
- #define DMA_CFG_BURSTPOWER_128 (7 << 8)
- #define DMA_CFG_BURSTPOWER_256 (8 << 8)
- #define DMA_CFG_BURSTPOWER_512 (9 << 8)
- #define DMA_CFG_BURSTPOWER_1024 (10 << 8)
- #define DMA_CFG_BURSTPOWER(n) ((n) << 8)
- #define DMA_CFG_SRCBURSTWRAP (1 << 14)
- #define DMA_CFG_DSTBURSTWRAP (1 << 15)
- #define DMA_CFG_CHPRIORITY(p) ((p) << 16)
- #define DMA_CTLSTAT_VALIDPENDING (1 << 0)
- #define DMA_CTLSTAT_TRIG (1 << 2)
- #define DMA_XFERCFG_CFGVALID (1 << 0)
- #define DMA_XFERCFG_RELOAD (1 << 1)
- #define DMA_XFERCFG_SWTRIG (1 << 2)

- #define DMA_XFERCFG_CLRTRIG (1 << 3)
- #define DMA_XFERCFG_SETINTA (1 << 4)
- #define DMA_XFERCFG_SETINTB (1 << 5)
- #define DMA_XFERCFG_WIDTH_8 (0 << 8)
- #define DMA_XFERCFG_WIDTH_16 (1 << 8)
- #define DMA_XFERCFG_WIDTH_32 (2 << 8)
- #define DMA_XFERCFG_SRCINC_0 (0 << 12)
- #define DMA_XFERCFG_SRCINC_1 (1 << 12)
- #define DMA_XFERCFG_SRCINC_2 (2 << 12)
- #define DMA_XFERCFG_SRCINC_4 (3 << 12)
- #define DMA_XFERCFG_DSTINC_0 (0 << 14)
- #define DMA_XFERCFG_DSTINC_1 (1 << 14)
- #define DMA_XFERCFG_DSTINC_2 (2 << 14)
- #define DMA_XFERCFG_DSTINC_4 (3 << 14)
- #define DMA_XFERCFG_XFERCOUNT(n) ((n - 1) << 16)

### Enumerations

- enum DMA_CHID_T {
  DMA_CH0, DMA_CH1, DMA_CH2, DMA_CH3,
  DMA_CH4, DMA_CH5, DMA_CH6, DMA_CH7,
  DMA_CH8, DMA_CH9, DMA_CH10, DMA_CH11,
  DMA_CH12, DMA_CH13, DMA_CH14, DMA_CH15,
  DMA_CH16, DMA_CH17, DMA_CH18, DMA_CH19,
  DMAREQ_FLEXCOMM0_RX = DMA_CH0, DMAREQ_FLEXCOMM0_TX, DMAREQ_FLEXCOMM1_RX, DMAREQ_FLEXCOMM1_TX,
  DMAREQ_FLEXCOMM2_RX, DMAREQ_FLEXCOMM2_TX, DMAREQ_FLEXCOMM3_RX, DMAREQ_FLEXCOMM3_TX,
  DMAREQ_FLEXCOMM4_RX, DMAREQ_FLEXCOMM4_TX, DMAREQ_FLEXCOMM5_RX, DMAREQ_FLEXCOMM5_TX,
  DMAREQ_FLEXCOMM6_RX, DMAREQ_FLEXCOMM6_TX, DMAREQ_FLEXCOMM7_RX, DMAREQ_FLEXCOMM7_TX,
  DMAREQ_DMIC0, DMAREQ_DMIC1, DMAREQ_SPIFI }

## 6.17.2 Macro Definition Documentation

### 6.17.2.1 #define DMA_ADDR( *addr* ) ((uint32_t) (addr))

Definition at line 149 of file dma_5411x.h.

### 6.17.2.2 #define DMA_CFG_BURSTPOWER( *n* ) ((n) << 8)

Set DMA burst size to 2^n transfers, max n=10

Definition at line 172 of file dma_5411x.h.

### 6.17.2.3 #define DMA_CFG_BURSTPOWER_1 (0 << 8)

Set DMA burst size to 1 transfer

Definition at line 161 of file dma_5411x.h.

#### 6.17.2.4 #define DMA_CFG_BURSTPOWER_1024 (10 << 8)

Set DMA burst size to 1024 transfers

Definition at line 171 of file dma_5411x.h.

#### 6.17.2.5 #define DMA_CFG_BURSTPOWER_128 (7 << 8)

Set DMA burst size to 128 transfers

Definition at line 168 of file dma_5411x.h.

#### 6.17.2.6 #define DMA_CFG_BURSTPOWER_16 (4 << 8)

Set DMA burst size to 16 transfers

Definition at line 165 of file dma_5411x.h.

#### 6.17.2.7 #define DMA_CFG_BURSTPOWER_2 (1 << 8)

Set DMA burst size to 2 transfers

Definition at line 162 of file dma_5411x.h.

#### 6.17.2.8 #define DMA_CFG_BURSTPOWER_256 (8 << 8)

Set DMA burst size to 256 transfers

Definition at line 169 of file dma_5411x.h.

#### 6.17.2.9 #define DMA_CFG_BURSTPOWER_32 (5 << 8)

Set DMA burst size to 32 transfers

Definition at line 166 of file dma_5411x.h.

#### 6.17.2.10 #define DMA_CFG_BURSTPOWER_4 (2 << 8)

Set DMA burst size to 4 transfers

Definition at line 163 of file dma_5411x.h.

#### 6.17.2.11 #define DMA_CFG_BURSTPOWER_512 (9 << 8)

Set DMA burst size to 512 transfers

Definition at line 170 of file dma_5411x.h.

#### 6.17.2.12 #define DMA_CFG_BURSTPOWER_64 (6 << 8)

Set DMA burst size to 64 transfers

Definition at line 167 of file dma_5411x.h.

#### 6.17.2.13 #define DMA_CFG_BURSTPOWER_8 (3 << 8)

Set DMA burst size to 8 transfers

Definition at line 164 of file dma_5411x.h.

#### 6.17.2.14 #define DMA_CFG_CHPRIORITY( *p* ) ((p) << 16)

Sets DMA channel priority, min 0 (highest), max 3 (lowest)

Definition at line 175 of file dma_5411x.h.

#### 6.17.2.15 #define DMA_CFG_DSTBURSTWRAP (1 << 15)

Destination burst wrapping is enabled for this DMA channel

Definition at line 174 of file dma_5411x.h.

#### 6.17.2.16 #define DMA_CFG_HWTRIGEN (1 << 1)

Use hardware triggering via imput mux

Definition at line 154 of file dma_5411x.h.

#### 6.17.2.17 #define DMA_CFG_PERIPHREQEN (1 << 0)

Enables Peripheral DMA requests

Definition at line 153 of file dma_5411x.h.

#### 6.17.2.18 #define DMA_CFG_SRCBURSTWRAP (1 << 14)

Source burst wrapping is enabled for this DMA channel

Definition at line 173 of file dma_5411x.h.

#### 6.17.2.19 #define DMA_CFG_TRIGBURST_BURST (1 << 6)

Burst transfer (see UM)

Definition at line 160 of file dma_5411x.h.

#### 6.17.2.20 #define DMA_CFG_TRIGBURST_SNGL (0 << 6)

Single transfer. Hardware trigger causes a single transfer

Definition at line 159 of file dma_5411x.h.

#### 6.17.2.21 #define DMA_CFG_TRIGPOL_HIGH (1 << 4)

Hardware trigger is active high or rising edge

Definition at line 156 of file dma_5411x.h.

#### 6.17.2.22 #define DMA_CFG_TRIGPOL_LOW (0 << 4)

Hardware trigger is active low or falling edge

Definition at line 155 of file dma_5411x.h.

#### 6.17.2.23 #define DMA_CFG_TRIGTYPE_EDGE (0 << 5)

Hardware trigger is edge triggered

Definition at line 157 of file dma_5411x.h.

#### 6.17.2.24 #define DMA_CFG_TRIGTYPE_LEVEL (1 << 5)

Hardware trigger is level triggered

Definition at line 158 of file dma_5411x.h.

#### 6.17.2.25 #define DMA_CTLSTAT_TRIG (1 << 2)

Trigger flag. Indicates that the trigger for this channel is currently set

Definition at line 179 of file dma_5411x.h.

#### 6.17.2.26 #define DMA_CTLSTAT_VALIDPENDING (1 << 0)

Valid pending flag for this channel

Definition at line 178 of file dma_5411x.h.

#### 6.17.2.27 #define DMA_INTSTAT_ACTIVEERRINT 0x4

Summarizes whether any error interrupts are pending

Definition at line 146 of file dma_5411x.h.

#### 6.17.2.28 #define DMA_INTSTAT_ACTIVEINT 0x2

Summarizes whether any enabled interrupts are pending

Definition at line 145 of file dma_5411x.h.

#### 6.17.2.29 #define DMA_XFERCFG_CFGVALID (1 << 0)

Configuration Valid flag

Definition at line 182 of file dma_5411x.h.

#### 6.17.2.30 #define DMA_XFERCFG_CLRTRIG (1 << 3)

Clear Trigger

Definition at line 185 of file dma_5411x.h.

#### 6.17.2.31 #define DMA_XFERCFG_DSTINC_0 (0 << 14)

DMA destination address is not incremented after a transfer

Definition at line 195 of file dma_5411x.h.

#### 6.17.2.32 #define DMA_XFERCFG_DSTINC_1 (1 << 14)

DMA destination address is incremented by 1 (width) after a transfer

Definition at line 196 of file dma_5411x.h.

#### 6.17.2.33 #define DMA_XFERCFG_DSTINC_2 (2 << 14)

DMA destination address is incremented by 2 (width) after a transfer

Definition at line 197 of file dma_5411x.h.

#### 6.17.2.34 #define DMA_XFERCFG_DSTINC_4 (3 << 14)

DMA destination address is incremented by 4 (width) after a transfer

Definition at line 198 of file dma_5411x.h.

#### 6.17.2.35 #define DMA_XFERCFG_RELOAD (1 << 1)

Indicates whether the channels control structure will be reloaded when the current descriptor is exhausted

Definition at line 183 of file dma_5411x.h.

#### 6.17.2.36 #define DMA_XFERCFG_SETINTA (1 << 4)

Set Interrupt flag A for this channel to fire when descriptor is complete

Definition at line 186 of file dma_5411x.h.

#### 6.17.2.37 #define DMA_XFERCFG_SETINTB (1 << 5)

Set Interrupt flag B for this channel to fire when descriptor is complete

Definition at line 187 of file dma_5411x.h.

#### 6.17.2.38 #define DMA_XFERCFG_SRCINC_0 (0 << 12)

DMA source address is not incremented after a transfer

Definition at line 191 of file dma_5411x.h.

#### 6.17.2.39 #define DMA_XFERCFG_SRCINC_1 (1 << 12)

DMA source address is incremented by 1 (width) after a transfer

Definition at line 192 of file dma_5411x.h.

#### 6.17.2.40 #define DMA_XFERCFG_SRCINC_2 (2 $<<$ 12)

DMA source address is incremented by 2 (width) after a transfer

Definition at line 193 of file dma_5411x.h.

#### 6.17.2.41 #define DMA_XFERCFG_SRCINC_4 (3 $<<$ 12)

DMA source address is incremented by 4 (width) after a transfer

Definition at line 194 of file dma_5411x.h.

#### 6.17.2.42 #define DMA_XFERCFG_SWTRIG (1 $<<$ 2)

Software Trigger

Definition at line 184 of file dma_5411x.h.

#### 6.17.2.43 #define DMA_XFERCFG_WIDTH_16 (1 $<<$ 8)

16-bit transfers are performed

Definition at line 189 of file dma_5411x.h.

#### 6.17.2.44 #define DMA_XFERCFG_WIDTH_32 (2 $<<$ 8)

32-bit transfers are performed

Definition at line 190 of file dma_5411x.h.

#### 6.17.2.45 #define DMA_XFERCFG_WIDTH_8 (0 $<<$ 8)

8-bit transfers are performed

Definition at line 188 of file dma_5411x.h.

#### 6.17.2.46 #define DMA_XFERCFG_XFERCOUNT( *n* ) ((n - 1) $<<$ 16)

DMA transfer count in 'transfers', between (0)1 and (1023)1024

Definition at line 199 of file dma_5411x.h.

#### 6.17.2.47 #define MAX_DMA_CHANNEL (20)

Definition at line 83 of file dma_5411x.h.

### 6.17.3 Enumeration Type Documentation

#### 6.17.3.1 enum DMA_CHID_T

**Enumerator**

***DMA_CH0***

***DMA_CH1***

***DMA_CH2***

***DMA_CH3***

***DMA_CH4***

***DMA_CH5***

***DMA_CH6***

***DMA_CH7***

***DMA_CH8***

***DMA_CH9***

***DMA_CH10***

***DMA_CH11***

***DMA_CH12***

***DMA_CH13***

***DMA_CH14***

***DMA_CH15***

***DMA_CH16***

***DMA_CH17***

***DMA_CH18***

***DMA_CH19***

***DMAREQ_FLEXCOMM0_RX***

***DMAREQ_FLEXCOMM0_TX***

***DMAREQ_FLEXCOMM1_RX***

***DMAREQ_FLEXCOMM1_TX***

***DMAREQ_FLEXCOMM2_RX***

***DMAREQ_FLEXCOMM2_TX***

***DMAREQ_FLEXCOMM3_RX***

***DMAREQ_FLEXCOMM3_TX***

***DMAREQ_FLEXCOMM4_RX***

***DMAREQ_FLEXCOMM4_TX***

***DMAREQ_FLEXCOMM5_RX***

***DMAREQ_FLEXCOMM5_TX***

***DMAREQ_FLEXCOMM6_RX***

***DMAREQ_FLEXCOMM6_TX***

***DMAREQ_FLEXCOMM7_RX***

***DMAREQ_FLEXCOMM7_TX***

***DMAREQ_DMIC0***

***DMAREQ_DMIC1***

***DMAREQ_SPIFI***

Definition at line 100 of file dma_5411x.h.

# 6.18 CHIP: LPC5411X DMA Service driver

## 6.18.1 Detailed Description

### Data Structures

- struct DMA_PERIPHERAL_CONTEXT_T
- struct DMA_DUAL_DESCRIPTOR_T

### Typedefs

- typedef void(∗ DMA_CALLBACK_T) (int32_t)

### Functions

- void Chip_DMASERVICE_Init (DMA_CHDESC_T ∗base)

  *Initialize DMA service.*
- void Chip_DMASERVICE_Isr (void)

  *DMA service interrupt handler.*
- void Chip_DMASERVICE_RegisterCb (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, DMA_CALLBACK_T pCallback)

  *Register callback function.*
- void Chip_DMASERVICE_SingleBuffer (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, uint32_t pMem, uint32_t length)

  *Use Single buffer mechanism.*
- void Chip_DMASERVICE_DoubleBuffer (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, uint32_t pMem, uint32_t length, DMA_DUAL_DESCRIPTOR_T ∗pD)

  *Use double buffer mechanism.*

### Variables

- uint32_t channel
- volatile uint32_t ∗ register_location
- uint32_t width
- uint32_t src_increment
- uint32_t dst_increment
- bool write
- DMA_CHDESC_T descr [2]

## 6.18.2 Typedef Documentation

### 6.18.2.1 typedef void(∗ DMA_CALLBACK_T) (int32_t)

Definition at line 44 of file dma_service_5411x.h.

## 6.18.3 Function Documentation

### 6.18.3.1 void Chip_DMASERVICE_DoubleBuffer ( const DMA_PERIPHERAL_CONTEXT_T ∗ *pContext,* uint32_t *pMem,* uint32_t *length,* DMA_DUAL_DESCRIPTOR_T ∗ *pD* )

Use double buffer mechanism.

**Parameters**

| | |
|---|---|
| *pContext* | : Pointer to peripheral channel context |
| *pMem* | : Pointer to data memory |
| *length* | : Transfer length in bytes |
| *pD* | : Pointer to dma dual descriptor |

**Returns**

Nothing

Definition at line 135 of file dma_service_5411x.c.

#### 6.18.3.2 void Chip_DMASERVICE_Init ( DMA_CHDESC_T ∗ *base* )

Initialize DMA service.

**Parameters**

| | |
|---|---|
| *base* | : The base address where the DMA descriptors will be stored |

**Returns**

Nothing

Definition at line 37 of file dma_service_5411x.c.

#### 6.18.3.3 void Chip_DMASERVICE_Isr ( void )

DMA service interrupt handler.

**Parameters**

| | |
|---|---|
| *None* | |

**Returns**

Nothing Must be called from DMA_IRQHandler

Definition at line 62 of file dma_service_5411x.c.

#### 6.18.3.4 void Chip_DMASERVICE_RegisterCb ( const DMA_PERIPHERAL_CONTEXT_T ∗ *pContext*, DMA_CALLBACK_T *pCallback* )

Register callback function.

**Parameters**

| | |
|---|---|
| *pContext* | : Pointer to peripheral channel context |
| *pCallback* | : Pointer to callback function |

**Returns**

Nothing

Definition at line 89 of file dma_service_5411x.c.

#### 6.18.3.5 void Chip_DMASERVICE_SingleBuffer ( const DMA_PERIPHERAL_CONTEXT_T * *pContext,* uint32_t *pMem,* uint32_t *length* )

Use Single buffer mechanism.

**Parameters**

| | |
|---|---|
| *pContext* | : Pointer to peripheral channel context |
| *pMem* | : Pointer to data memory |
| *length* | : Transfer length in bytes |

**Returns**

Nothing

Definition at line 99 of file dma_service_5411x.c.

## 6.18.4 Variable Documentation

### 6.18.4.1 uint32_t channel

Definition at line 47 of file dma_service_5411x.h.

### 6.18.4.2 DMA_CHDESC_T descr[2]

Definition at line 56 of file dma_service_5411x.h.

### 6.18.4.3 uint32_t dst_increment

Definition at line 51 of file dma_service_5411x.h.

### 6.18.4.4 volatile uint32_t* register_location

Definition at line 48 of file dma_service_5411x.h.

### 6.18.4.5 uint32_t src_increment

Definition at line 50 of file dma_service_5411x.h.

### 6.18.4.6 uint32_t width

Definition at line 49 of file dma_service_5411x.h.

### 6.18.4.7 bool write

Definition at line 52 of file dma_service_5411x.h.

## 6.19 CHIP: LPC5411X DMIC driver

### 6.19.1 Detailed Description

#### Data Structures

- struct LPC_DMIC_Channel_Type
- struct LPC_DMIC_T
- struct DMIC_STATISTICS_T

  *DMIC statistics structure.*
- struct DMIC_CHANNEL_CONFIG_T

#### Macros

- #define DMIC_FIFO_ENABLE_P 0
- #define DMIC_FIFO_RESETN_P 1
- #define DMIC_FIFO_INTREN_P 2
- #define DMIC_FIFO_DMAEN_P 3
- #define DMIC_FIFO_TLVL_P 16
- #define DMIC_FIFO_ENABLE (1<<DMIC_FIFO_ENABLE_P)
- #define DMIC_FIFO_RESETN (1<<DMIC_FIFO_RESETN_P)
- #define DMIC_FIFO_INTREN (1<<DMIC_FIFO_INTREN_P)
- #define DMIC_FIFO_DMAEN (1<<DMIC_FIFO_DMAEN_P )
- #define DMIC_FIFO_INT_P 0
- #define DMIC_FIFO_OVERRUN_P 1
- #define DMIC_FIFO_UNDERRUN_P 2
- #define DMIC_FIFO_INT (1<<DMIC_FIFO_INT_P )
- #define DMIC_FIFO_OVERRUN (1<<DMIC_FIFO_OVERRUN_P )
- #define DMIC_FIFO_UNDERRUN (1<<DMIC_FIFO_UNDERRUN_P)
- #define DMIC_PHY_FALL_P 0
- #define DMIC_PHY_HALF_P 1
- #define DMIC_PHY_FALL (1<<DMIC_PHY_FALL_P)
- #define DMIC_PHY_HALF (1<<DMIC_PHY_HALF_P)
- #define DMIC_DCPOLE_P 0
- #define DMIC_DCGAIN_REDUCE_P 4
- #define DMIC_SATURATE_AT16BIT_P 8

#### Enumerations

- enum OP_MODE_T { DMIC_OP_POLL, DMIC_OP_INTR, DMIC_OP_DMA }
- enum STEREO_SIDE_T { DMIC_LEFT = 0, DMIC_RIGHT = 1 }
- enum PDM_DIV_T {
  DMIC_PDM_DIV1 = 0, DMIC_PDM_DIV2 = 1, DMIC_PDM_DIV3 = 2, DMIC_PDM_DIV4 = 3,
  DMIC_PDM_DIV6 = 4, DMIC_PDM_DIV8 = 5, DMIC_PDM_DIV12 = 6, DMIC_PDM_DIV16 = 7,
  DMIC_PDM_DIV24 = 8, DMIC_PDM_DIV32 = 9, DMIC_PDM_DIV48 = 10, DMIC_PDM_DIV64 = 11,
  DMIC_PDM_DIV96 = 12, DMIC_PDM_DIV128 = 13 }
- enum COMPENSATION_T { DMIC_COMP0_0 = 0, DMIC_COMP0_16 = 1, DMIC_COMP0_15 = 2, DMIC_COMP0_13 = 3 }
- enum DC_REMOVAL_T { DMIC_DC_NOREMOVE = 0, DMIC_DC_CUT155 = 1, DMIC_DC_CUT78 = 2, DMIC_DC_CUT39 = 3 }
- enum DMIC_IO_T {
  pdm_dual = 0, pdm_stereo = 4, pdm_bypass = 3, pdm_bypass_clk0 = 1,
  pdm_bypass_clk1 = 2 }

## Functions

- void Chip_DMIC_Init (const CHIP_SYSCON_CLOCK_T clock, const CHIP_SYSCON_PERIPH_RESET_T reset)

  *Initialize DMIC interface.*
- void Chip_DMIC_CfgIO (LPC_DMIC_T ∗pDMIC, DMIC_IO_T cfg)

  *Configure DMIC io.*
- void Chip_DMIC_SetOpMode (LPC_DMIC_T ∗pDMIC, OP_MODE_T mode)

  *Set DMIC operating mode.*
- void Chip_DMIC_CfgChannel (LPC_DMIC_T ∗pDMIC, uint32_t channel, DMIC_CHANNEL_CONFIG_↩ T ∗channel_cfg)

  *Configure DMIC channel.*
- void Chip_DMIC_CfgChannelDc (LPC_DMIC_T ∗pDMIC, uint32_t channel, DC_REMOVAL_T dc_cut_level, uint32_t post_dc_gain_reduce, bool saturate16bit)

  *Configure DMIC channel DC removal setting.*
- void Chip_DMIC_Use2fs (LPC_DMIC_T ∗pDMIC, bool use2fs)

  *Configure Clock scaling.*
- void Chip_DMIC_EnableChannnel (LPC_DMIC_T ∗pDMIC, uint32_t channelmask)

  *Configure Clock scaling.*
- void Chip_DMIC_FifoChannel (LPC_DMIC_T ∗pDMIC, uint32_t channel, uint32_t trig_level, uint32_t enable, uint32_t resetn)

  *Configure fifo settings for DMIC channel.*
- __STATIC_INLINE uint32_t Chip_DMIC_FifoGetStatus (LPC_DMIC_T ∗pDMIC, uint32_t channel)

  *Get FIFO status.*
- __STATIC_INLINE void Chip_DMIC_FifoClearStatus (LPC_DMIC_T ∗pDMIC, uint32_t channel, uint32_↩ t mask)

  *Clear FIFO status.*
- __STATIC_INLINE uint32_t Chip_DMIC_FifoGetData (LPC_DMIC_T ∗pDMIC, uint32_t channel)

  *Get FIFO data.*

## Variables

- __IO uint32_t OSR
- __IO uint32_t DIVHFCLK
- __IO uint32_t PREAC2FSCOEF
- __IO uint32_t PREAC4FSCOEF
- __IO uint32_t GAINSHFT
- __IO uint32_t TDM96EN
- __IO uint32_t TDM19EN
- __IO uint32_t reserved [25]
- __IO uint32_t FIFO_CTRL
- __IO uint32_t FIFO_STATUS
- __IO uint32_t FIFO_DATA
- __IO uint32_t PHY_CTRL
- __IO uint32_t DC_CTRL
- __IO uint32_t reserved1 [27]
- __IO LPC_DMIC_Channel_Type CHANNEL [15]
- __IO uint32_t CHANEN
- __IO uint32_t reserved0 [2]
- __IO uint32_t IOCFG
- __IO uint32_t USE2FS
- __IO uint32_t reserved [27]
- __IO uint32_t HWVADGAIN

- __IO uint32_t HWVADHPFS
- __IO uint32_t HWVADST10
- __IO uint32_t HWVADRSTT
- __IO uint32_t HWVADTHGN
- __IO uint32_t HWVADTHGS
- __IO uint32_t HWVADLOWZ
- __IO uint32_t reserved1 [24]
- __O uint32_t ID
- uint32_t fifo_ints
- uint32_t fifo_overrun
- uint32_t fifo_underrun
- STEREO_SIDE_T side
- PDM_DIV_T divhfclk
- uint32_t osr
- int32_t gainshft
- COMPENSATION_T preac2coef
- COMPENSATION_T preac4coef
- DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_interleaved_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_interleaved_context

## 6.19.2 Macro Definition Documentation

### 6.19.2.1 #define DMIC_DCGAIN_REDUCE_P 4

Definition at line 137 of file dmic_5411x.h.

### 6.19.2.2 #define DMIC_DCPOLE_P 0

Definition at line 136 of file dmic_5411x.h.

### 6.19.2.3 #define DMIC_FIFO_DMAEN (1<<DMIC_FIFO_DMAEN_P )

Definition at line 116 of file dmic_5411x.h.

### 6.19.2.4 #define DMIC_FIFO_DMAEN_P 3

Definition at line 109 of file dmic_5411x.h.

### 6.19.2.5 #define DMIC_FIFO_ENABLE (1<<DMIC_FIFO_ENABLE_P)

Definition at line 113 of file dmic_5411x.h.

### 6.19.2.6 #define DMIC_FIFO_ENABLE_P 0

Definition at line 106 of file dmic_5411x.h.

### 6.19.2.7 #define DMIC_FIFO_INT (1<<DMIC_FIFO_INT_P )

Definition at line 123 of file dmic_5411x.h.

#### 6.19.2.8 #define DMIC_FIFO_INT_P 0

Definition at line 119 of file dmic_5411x.h.

#### 6.19.2.9 #define DMIC_FIFO_INTREN (1<<DMIC_FIFO_INTREN_P)

Definition at line 115 of file dmic_5411x.h.

#### 6.19.2.10 #define DMIC_FIFO_INTREN_P 2

Definition at line 108 of file dmic_5411x.h.

#### 6.19.2.11 #define DMIC_FIFO_OVERRUN (1<<DMIC_FIFO_OVERRUN_P )

Definition at line 124 of file dmic_5411x.h.

#### 6.19.2.12 #define DMIC_FIFO_OVERRUN_P 1

Definition at line 120 of file dmic_5411x.h.

#### 6.19.2.13 #define DMIC_FIFO_RESETN (1<<DMIC_FIFO_RESETN_P)

Definition at line 114 of file dmic_5411x.h.

#### 6.19.2.14 #define DMIC_FIFO_RESETN_P 1

Definition at line 107 of file dmic_5411x.h.

#### 6.19.2.15 #define DMIC_FIFO_TLVL_P 16

Definition at line 111 of file dmic_5411x.h.

#### 6.19.2.16 #define DMIC_FIFO_UNDERRUN (1<<DMIC_FIFO_UNDERRUN_P)

Definition at line 125 of file dmic_5411x.h.

#### 6.19.2.17 #define DMIC_FIFO_UNDERRUN_P 2

Definition at line 121 of file dmic_5411x.h.

#### 6.19.2.18 #define DMIC_PHY_FALL (1<<DMIC_PHY_FALL_P)

Definition at line 131 of file dmic_5411x.h.

#### 6.19.2.19 #define DMIC_PHY_FALL_P 0

Definition at line 128 of file dmic_5411x.h.

#### 6.19.2.20 #define DMIC_PHY_HALF (1<<DMIC_PHY_HALF_P)

Definition at line 132 of file dmic_5411x.h.

#### 6.19.2.21 #define DMIC_PHY_HALF_P 1

Definition at line 129 of file dmic_5411x.h.

#### 6.19.2.22 #define DMIC_SATURATE_AT16BIT_P 8

Definition at line 138 of file dmic_5411x.h.

### 6.19.3 Enumeration Type Documentation

#### 6.19.3.1 enum COMPENSATION_T

**Enumerator**

***DMIC_COMP0_0***

***DMIC_COMP0_16***

***DMIC_COMP0_15***

***DMIC_COMP0_13***

Definition at line 170 of file dmic_5411x.h.

#### 6.19.3.2 enum DC_REMOVAL_T

**Enumerator**

***DMIC_DC_NOREMOVE***

***DMIC_DC_CUT155***

***DMIC_DC_CUT78***

***DMIC_DC_CUT39***

Definition at line 177 of file dmic_5411x.h.

#### 6.19.3.3 enum DMIC_IO_T

**Enumerator**

***pdm_dual*** Two separate pairs of PDM wires

***pdm_stereo*** Stereo Mic

***pdm_bypass*** Clk Bypas clocks both channels

***pdm_bypass_clk0*** Clk Bypas clocks only channel0

***pdm_bypass_clk1*** Clk Bypas clocks only channel1

Definition at line 184 of file dmic_5411x.h.

#### 6.19.3.4 enum OP_MODE_T

**Enumerator**

***DMIC_OP_POLL***

***DMIC_OP_INTR***

***DMIC_OP_DMA***

Definition at line 142 of file dmic_5411x.h.

#### 6.19.3.5 enum PDM_DIV_T

**Enumerator**

***DMIC_PDM_DIV1***

***DMIC_PDM_DIV2***

***DMIC_PDM_DIV3***

***DMIC_PDM_DIV4***

***DMIC_PDM_DIV6***

***DMIC_PDM_DIV8***

***DMIC_PDM_DIV12***

***DMIC_PDM_DIV16***

***DMIC_PDM_DIV24***

***DMIC_PDM_DIV32***

***DMIC_PDM_DIV48***

***DMIC_PDM_DIV64***

***DMIC_PDM_DIV96***

***DMIC_PDM_DIV128***

Definition at line 153 of file dmic_5411x.h.

#### 6.19.3.6 enum STEREO_SIDE_T

**Enumerator**

***DMIC_LEFT***

***DMIC_RIGHT***

Definition at line 148 of file dmic_5411x.h.

### 6.19.4 Function Documentation

#### 6.19.4.1 void Chip_DMIC_CfgChannel ( LPC_DMIC_T ∗ *pDMIC,* uint32_t *channel,* DMIC_CHANNEL_CONFIG_T ∗ *channel_cfg* )

Configure DMIC channel.

**Parameters**

| | |
|---|---|
| *pDMIC* | : The base address of DMIC interface |
| *channel* | : DMIC channel |
| *channel_cfg* | : Channel configuration |

**Returns**

Nothing

Definition at line 109 of file dmic_5411x.c.

#### 6.19.4.2 void Chip_DMIC_CfgChannelDc ( LPC_DMIC_T ∗ *pDMIC,* uint32_t *channel,* DC_REMOVAL_T *dc_cut_level,* uint32_t *post_dc_gain_reduce,* bool *saturate16bit* )

Configure DMIC channel DC removal setting.

**Parameters**

| | |
|---|---|
| *pDMIC* | : The base address of DMIC interface |
| *channel* | : DMIC channel |
| *dc_cut_level* | : DC cut level |
| *post_dc_gain_←reduce* | : Post DC cut gain adjustment |
| *saturate16bit* | : Saturation setting |

**Returns**

Nothing

Definition at line 120 of file dmic_5411x.c.

#### 6.19.4.3 void Chip_DMIC_CfgIO ( LPC_DMIC_T ∗ *pDMIC,* DMIC_IO_T *cfg* )

Configure DMIC io.

**Parameters**

| | |
|---|---|
| *pDMIC* | : The base address of DMIC interface |
| *cfg* | : DMIC io configuration |

**Returns**

Nothing

Definition at line 86 of file dmic_5411x.c.

#### 6.19.4.4 void Chip_DMIC_EnableChannnel ( LPC_DMIC_T ∗ *pDMIC,* uint32_t *channelmask* )

Configure Clock scaling.

**Parameters**

| | |
|---|---|
| *pDMIC* | : The base address of DMIC interface |
| *channelmask* | : Channel selection |

**Returns**

Nothing

Definition at line 132 of file dmic_5411x.c.

#### 6.19.4.5 void Chip_DMIC_FifoChannel ( LPC_DMIC_T ∗ *pDMIC,* uint32_t *channel,* uint32_t *trig_level,* uint32_t *enable,* uint32_t *resetn* )

Configure fifo settings for DMIC channel.

**Parameters**

| | |
|---:|---|
| *pDMIC* | : The base address of DMIC interface |
| *channel* | : DMIC channel |
| *trig_level* | : FIFO trigger level |
| *enable* | : FIFO level |
| *resetn* | : FIFO reset |

**Returns**

Nothing

Definition at line 138 of file dmic_5411x.c.

#### 6.19.4.6 __STATIC_INLINE void Chip_DMIC_FifoClearStatus ( LPC_DMIC_T ∗ *pDMIC,* uint32_t *channel,* uint32_t *mask* )

Clear FIFO status.

**Parameters**

| | |
|---:|---|
| *pDMIC* | : The base address of DMIC interface |
| *channel* | : DMIC channel |
| *mask* | : Bits to be cleared |

**Returns**

FIFO status

Definition at line 290 of file dmic_5411x.h.

#### 6.19.4.7 __STATIC_INLINE uint32_t Chip_DMIC_FifoGetData ( LPC_DMIC_T ∗ *pDMIC,* uint32_t *channel* )

Get FIFO data.

**Parameters**

| | |
|---:|---|
| *pDMIC* | : The base address of DMIC interface |
| *channel* | : DMIC channel |

**Returns**

FIFO data

Definition at line 301 of file dmic_5411x.h.

#### 6.19.4.8 __STATIC_INLINE uint32_t Chip_DMIC_FifoGetStatus ( LPC_DMIC_T ∗ *pDMIC,* uint32_t *channel* )

Get FIFO status.

**Parameters**

| | |
|---:|---|
| *pDMIC* | : The base address of DMIC interface |
| *channel* | : DMIC channel |

**Returns**

FIFO status

Definition at line 278 of file dmic_5411x.h.

#### 6.19.4.9 void Chip_DMIC_Init ( const CHIP_SYSCON_CLOCK_T *clock*, const CHIP_SYSCON_PERIPH_RESET_T *reset* )

Initialize DMIC interface.

**Parameters**

| | |
|---|---|
| *clock* | : DMIC clock assignment |
| *reset* | : DMIC reset assignment |

**Returns**

Nothing

Definition at line 76 of file dmic_5411x.c.

#### 6.19.4.10 void Chip_DMIC_SetOpMode ( LPC_DMIC_T ∗ *pDMIC,* OP_MODE_T *mode* )

Set DMIC operating mode.

**Parameters**

| | |
|---|---|
| *pDMIC* | : The base address of DMIC interface |
| *mode* | : DMIC mode |

**Returns**

Nothing

Definition at line 92 of file dmic_5411x.c.

#### 6.19.4.11 void Chip_DMIC_Use2fs ( LPC_DMIC_T ∗ *pDMIC,* bool *use2fs* )

Configure Clock scaling.

**Parameters**

| | |
|---|---|
| *pDMIC* | : The base address of DMIC interface |
| *use2fs* | : clock scaling |

**Returns**

Nothing

Definition at line 126 of file dmic_5411x.c.

### 6.19.5 Variable Documentation

#### 6.19.5.1 __IO uint32_t CHANEN

Definition at line 65 of file dmic_5411x.h.

#### 6.19.5.2 __IO LPC_DMIC_Channel_Type CHANNEL[15]

Definition at line 63 of file dmic_5411x.h.

#### 6.19.5.3 __IO uint32_t DC_CTRL

Definition at line 57 of file dmic_5411x.h.

#### 6.19.5.4 __IO uint32_t DIVHFCLK

Definition at line 46 of file dmic_5411x.h.

#### 6.19.5.5 PDM_DIV_T divhfclk

Definition at line 194 of file dmic_5411x.h.

#### 6.19.5.6 DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_context

Definition at line 35 of file dmic_5411x.c.

#### 6.19.5.7 DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_interleaved_context

Definition at line 56 of file dmic_5411x.c.

#### 6.19.5.8 DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_context

Definition at line 45 of file dmic_5411x.c.

#### 6.19.5.9 DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_interleaved_context

Definition at line 66 of file dmic_5411x.c.

#### 6.19.5.10 __IO uint32_t FIFO_CTRL

Definition at line 53 of file dmic_5411x.h.

#### 6.19.5.11 __IO uint32_t FIFO_DATA

Definition at line 55 of file dmic_5411x.h.

#### 6.19.5.12 uint32_t fifo_ints

count: FIFO interrupts

Definition at line 88 of file dmic_5411x.h.

#### 6.19.5.13 uint32_t fifo_overrun

count: FIFO over-run errors

Definition at line 89 of file dmic_5411x.h.

#### 6.19.5.14 __IO uint32_t FIFO_STATUS

Definition at line 54 of file dmic_5411x.h.

#### 6.19.5.15 uint32_t fifo_underrun

count: FIFO under-run errors

Definition at line 90 of file dmic_5411x.h.

#### 6.19.5.16 __IO uint32_t GAINSHFT

Definition at line 49 of file dmic_5411x.h.

#### 6.19.5.17 int32_t gainshft

Definition at line 196 of file dmic_5411x.h.

#### 6.19.5.18 __IO uint32_t HWVADGAIN

Definition at line 71 of file dmic_5411x.h.

#### 6.19.5.19 __IO uint32_t HWVADHPFS

Definition at line 72 of file dmic_5411x.h.

#### 6.19.5.20 __IO uint32_t HWVADLOWZ

Definition at line 77 of file dmic_5411x.h.

#### 6.19.5.21 __IO uint32_t HWVADRSTT

Definition at line 74 of file dmic_5411x.h.

#### 6.19.5.22 __IO uint32_t HWVADST10

Definition at line 73 of file dmic_5411x.h.

#### 6.19.5.23 __IO uint32_t HWVADTHGN

Definition at line 75 of file dmic_5411x.h.

#### 6.19.5.24 __IO uint32_t HWVADTHGS

Definition at line 76 of file dmic_5411x.h.

#### 6.19.5.25 __O uint32_t ID

Definition at line 79 of file dmic_5411x.h.

#### 6.19.5.26 __IO uint32_t IOCFG

Definition at line 67 of file dmic_5411x.h.

#### 6.19.5.27 __IO uint32_t OSR

Definition at line 45 of file dmic_5411x.h.

#### 6.19.5.28 uint32_t osr

Definition at line 195 of file dmic_5411x.h.

#### 6.19.5.29 __IO uint32_t PHY_CTRL

Definition at line 56 of file dmic_5411x.h.

#### 6.19.5.30 COMPENSATION_T preac2coef

Definition at line 197 of file dmic_5411x.h.

#### 6.19.5.31 __IO uint32_t PREAC2FSCOEF

Definition at line 47 of file dmic_5411x.h.

#### 6.19.5.32 COMPENSATION_T preac4coef

Definition at line 198 of file dmic_5411x.h.

#### 6.19.5.33 __IO uint32_t PREAC4FSCOEF

Definition at line 48 of file dmic_5411x.h.

#### 6.19.5.34 __IO uint32_t reserved[25]

Definition at line 52 of file dmic_5411x.h.

#### 6.19.5.35 __IO uint32_t reserved[27]

Definition at line 69 of file dmic_5411x.h.

#### 6.19.5.36 __IO uint32_t reserved0[2]

Definition at line 66 of file dmic_5411x.h.

#### 6.19.5.37 __IO uint32_t reserved1[27]

Definition at line 58 of file dmic_5411x.h.

#### 6.19.5.38 __IO uint32_t reserved1[24]

Definition at line 78 of file dmic_5411x.h.

#### 6.19.5.39 STEREO_SIDE_T side

Definition at line 193 of file dmic_5411x.h.

#### 6.19.5.40 __IO uint32_t TDM19EN

Definition at line 51 of file dmic_5411x.h.

#### 6.19.5.41 __IO uint32_t TDM96EN

Definition at line 50 of file dmic_5411x.h.

#### 6.19.5.42 __IO uint32_t USE2FS

Definition at line 68 of file dmic_5411x.h.

## 6.20 CHIP: LPC5411X Enhanced boot block support

### 6.20.1 Detailed Description

### Data Structures

- struct PINTABLE_T

    *LPC5411X Pin table structure used for enhanced boot block support.*

### Macros

- #define IMAGE_ENH_MARKER_OFF 0x24
- #define IMAGE_SINGLE_ENH_SIG 0xEDDC9494
- #define IMAGE_DUAL_ENH_SIG 0x0FFEB6B6
- #define IMAGE_BOOT_BLOCK_OFF 0x28
- #define IMAGE_ENH_BLOCK_MARKER 0xFEEDA5A5
- #define SETPORTPIN(port, pin) (((port) & 0x7) << 5) | ((pin) & 0x1F)

### Enumerations

- enum IMAGE_T {
  IMG_NORMAL = 0, IMG_ISP_WAIT, IMG_NO_WAIT, IMG_NO_CRC,
  IMG_JUST_BOOT = 0xFF }
- enum IFSEL_T {
  SL_AUTO = 0, SL_I2C0, SL_I2C1, SL_I2C2,
  SL_SPI0, SL_SPI1 }

### 6.20.2 Macro Definition Documentation

#### 6.20.2.1 #define IMAGE_BOOT_BLOCK_OFF 0x28

Definition at line 54 of file pintable_5411x.h.

#### 6.20.2.2 #define IMAGE_DUAL_ENH_SIG 0x0FFEB6B6

Definition at line 51 of file pintable_5411x.h.

#### 6.20.2.3 #define IMAGE_ENH_BLOCK_MARKER 0xFEEDA5A5

Definition at line 57 of file pintable_5411x.h.

#### 6.20.2.4 #define IMAGE_ENH_MARKER_OFF 0x24

Definition at line 45 of file pintable_5411x.h.

#### 6.20.2.5 #define IMAGE_SINGLE_ENH_SIG 0xEDDC9494

Definition at line 48 of file pintable_5411x.h.

#### 6.20.2.6 #define SETPORTPIN( *port*, *pin* ) (((port) & 0x7) << 5) | ((pin) & 0x1F)

Definition at line 60 of file pintable_5411x.h.

### 6.20.3 Enumeration Type Documentation

#### 6.20.3.1 enum IFSEL_T

Host interface sources (ifSel) for the pin table structure

**Enumerator**

***SL_AUTO*** Auto-detect used for host interface

***SL_I2C0*** I2C0 used for host interface

***SL_I2C1*** I2C1 used for host interface

***SL_I2C2*** I2C2 used for host interface

***SL_SPI0*** SPI0 used for host interface

***SL_SPI1*** SPI1 used for host interface

Definition at line 72 of file pintable_5411x.h.

#### 6.20.3.2 enum IMAGE_T

Image type (img_type) for the pin table structure

**Enumerator**

***IMG_NORMAL*** Normal image check, assert dynamic ISP to halt boot

***IMG_ISP_WAIT*** Wait for host system to send SH_CMD_BOOT command

***IMG_NO_WAIT*** Boot image without checking dynamic ISP, CRC is still done

***IMG_NO_CRC*** No CRC check made. Used during development. Dynamic ISP still works

***IMG_JUST_BOOT*** Disables XOR and CRC checks, will always boot

Definition at line 63 of file pintable_5411x.h.

## 6.21 CHIP: LPC5411X GPIO driver

### 6.21.1 Detailed Description

#### Data Structures

- struct LPC_GPIO_T

  *GPIO port register block structure.*

#### Functions

- __STATIC_INLINE void Chip_GPIO_Init (LPC_GPIO_T ∗pGPIO)

  *Initialize GPIO block.*
- __STATIC_INLINE void Chip_GPIO_DeInit (LPC_GPIO_T ∗pGPIO)

  *De-Initialize GPIO block.*
- __STATIC_INLINE void Chip_GPIO_WritePortBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t pin, bool setting)

  *Set a GPIO port/pin state.*
- __STATIC_INLINE void Chip_GPIO_SetPinState (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin, bool setting)

  *Set a GPIO pin state via the GPIO byte register.*
- __STATIC_INLINE bool Chip_GPIO_ReadPortBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t pin)

  *Read a GPIO pin state via the GPIO byte register.*
- __STATIC_INLINE bool Chip_GPIO_GetPinState (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Get a GPIO pin state via the GPIO byte register.*
- __STATIC_INLINE void Chip_GPIO_WriteDirBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t pin, bool setting)

  *Set GPIO direction for a single GPIO pin.*
- __STATIC_INLINE void Chip_GPIO_SetPinDIROutput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set GPIO direction for a single GPIO pin to an output.*
- __STATIC_INLINE void Chip_GPIO_SetPinDIRInput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set GPIO direction for a single GPIO pin to an input.*
- __STATIC_INLINE void Chip_GPIO_SetPinDIR (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin, bool output)

  *Set GPIO direction for a single GPIO pin.*
- __STATIC_INLINE bool Chip_GPIO_ReadDirBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t bit)

  *Read a GPIO direction (out or in)*
- __STATIC_INLINE bool Chip_GPIO_GetPinDIR (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Get GPIO direction for a single GPIO pin.*
- __STATIC_INLINE void Chip_GPIO_SetDir (LPC_GPIO_T ∗pGPIO, uint8_t portNum, uint32_t bitValue, uint8_t out)

  *Set Direction for a GPIO port.*
- __STATIC_INLINE void Chip_GPIO_SetPortDIROutput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pin↩Mask)

  *Set GPIO direction for a all selected GPIO pins to an output.*
- __STATIC_INLINE void Chip_GPIO_SetPortDIRInput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pin↩Mask)

  *Set GPIO direction for a all selected GPIO pins to an input.*
- __STATIC_INLINE void Chip_GPIO_SetPortDIR (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pinMask, bool outSet)

  *Set GPIO direction for a all selected GPIO pins to an input or output.*
- __STATIC_INLINE uint32_t Chip_GPIO_GetPortDIR (LPC_GPIO_T ∗pGPIO, uint8_t port)

*Get GPIO direction for a all GPIO pins.*

- __STATIC_INLINE void Chip_GPIO_SetPortMask (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t mask)

  *Set GPIO port mask value for GPIO masked read and write.*
- __STATIC_INLINE uint32_t Chip_GPIO_GetPortMask (LPC_GPIO_T ∗pGPIO, uint8_t port)

  *Get GPIO port mask value used for GPIO masked read and write.*
- __STATIC_INLINE void Chip_GPIO_SetPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t value)

  *Set all GPIO raw pin states (regardless of masking)*
- __STATIC_INLINE uint32_t Chip_GPIO_GetPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port)

  *Get all GPIO raw pin states (regardless of masking)*
- __STATIC_INLINE void Chip_GPIO_SetMaskedPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t value)

  *Set all GPIO pin states, but mask via the MASKP0 register.*
- __STATIC_INLINE uint32_t Chip_GPIO_GetMaskedPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port)

  *Get all GPIO pin statesm but mask via the MASKP0 register.*
- __STATIC_INLINE void Chip_GPIO_SetValue (LPC_GPIO_T ∗pGPIO, uint8_t portNum, uint32_t bitValue)

  *Set a GPIO port/bit to the high state.*
- __STATIC_INLINE void Chip_GPIO_SetPortOutHigh (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pins)

  *Set selected GPIO output pins to the high state.*
- __STATIC_INLINE void Chip_GPIO_SetPinOutHigh (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set an individual GPIO output pin to the high state.*
- __STATIC_INLINE void Chip_GPIO_ClearValue (LPC_GPIO_T ∗pGPIO, uint8_t portNum, uint32_t bitValue)

  *Set a GPIO port/bit to the low state.*
- __STATIC_INLINE void Chip_GPIO_SetPortOutLow (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pins)

  *Set selected GPIO output pins to the low state.*
- __STATIC_INLINE void Chip_GPIO_SetPinOutLow (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set an individual GPIO output pin to the low state.*
- __STATIC_INLINE void Chip_GPIO_SetPortToggle (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pins)

  *Toggle selected GPIO output pins to the opposite state.*
- __STATIC_INLINE void Chip_GPIO_SetPinToggle (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Toggle an individual GPIO output pin to the opposite state.*
- __STATIC_INLINE uint32_t Chip_GPIO_ReadValue (LPC_GPIO_T ∗pGPIO, uint8_t portNum)

  *Read current bit states for the selected port.*

## 6.21.2 Function Documentation

### 6.21.2.1 __STATIC_INLINE void Chip_GPIO_ClearValue ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *portNum,* uint32_t *bitValue* )

Set a GPIO port/bit to the low state.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *portNum* | : port number (support port 0 only) |
| *bitValue* | : bit(s) in the port to set low |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output. It is recommended to use the Chip_GPIO_SetPortOutLow() function instead.

Definition at line 432 of file gpio_5411x.h.

#### 6.21.2.2 __STATIC_INLINE void Chip_GPIO_DeInit ( LPC_GPIO_T ∗ *pGPIO* )

De-Initialize GPIO block.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip (not used) |

**Returns**

Nothing

Definition at line 79 of file gpio_5411x.h.

#### 6.21.2.3 __STATIC_INLINE uint32_t Chip_GPIO_GetMaskedPortValue ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port* )

Get all GPIO pin statesm but mask via the MASKP0 register.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |

**Returns**

Current (masked) state of all GPIO pins

Definition at line 374 of file gpio_5411x.h.

#### 6.21.2.4 __STATIC_INLINE bool Chip_GPIO_GetPinDIR ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint8_t *pin* )

Get GPIO direction for a single GPIO pin.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to read (supports port 0 only) |
| *pin* | : GPIO pin to get direction for |

**Returns**

true if the GPIO is an output, false if input

Definition at line 222 of file gpio_5411x.h.

#### 6.21.2.5 __STATIC_INLINE bool Chip_GPIO_GetPinState ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint8_t *pin* )

Get a GPIO pin state via the GPIO byte register.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to read |
| *pin* | : GPIO pin to get state for |

**Returns**

true if the GPIO is high, false if low

**Note**

This function replaces Chip_GPIO_ReadPortBit()

Definition at line 134 of file gpio_5411x.h.

#### 6.21.2.6 __STATIC_INLINE uint32_t Chip_GPIO_GetPortDIR ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port* )

Get GPIO direction for a all GPIO pins.

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number |

**Returns**

a bitfield containing the input and output states for each pin

**Note**

For pins 0..n, a high state in a bit corresponds to an output state for the same pin, while a low state corresponds to an input state.

Definition at line 303 of file gpio_5411x.h.

#### 6.21.2.7 __STATIC_INLINE uint32_t Chip_GPIO_GetPortMask ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port* )

Get GPIO port mask value used for GPIO masked read and write.

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |

**Returns**

Returns value set with the Chip_GPIO_SetPortMask() function.

Definition at line 328 of file gpio_5411x.h.

#### 6.21.2.8 __STATIC_INLINE uint32_t Chip_GPIO_GetPortValue ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port* )

Get all GPIO raw pin states (regardless of masking)

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |

**Returns**

Current (raw) state of all GPIO pins

Definition at line 351 of file gpio_5411x.h.

#### 6.21.2.9 __STATIC_INLINE void Chip_GPIO_Init ( LPC_GPIO_T ∗ *pGPIO* )

Initialize GPIO block.

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip (not used) |

**Returns**

Nothing

**Note**

*pGPIO* parameter is ignored and all available GPIOs will be turned on by default.

Definition at line 66 of file gpio_5411x.h.

### 6.21.2.10 __STATIC_INLINE bool Chip_GPIO_ReadDirBit ( LPC_GPIO_T ∗ *pGPIO*, uint32_t *port*, uint8_t *bit* )

Read a GPIO direction (out or in)

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to read |
| *bit* | : GPIO bit direction to read |

**Returns**

true if the GPIO is an output, false if input

**Note**

It is recommended to use the Chip_GPIO_GetPinDIR() function instead.

Definition at line 210 of file gpio_5411x.h.

### 6.21.2.11 __STATIC_INLINE bool Chip_GPIO_ReadPortBit ( LPC_GPIO_T ∗ *pGPIO*, uint32_t *port*, uint8_t *pin* )

Read a GPIO pin state via the GPIO byte register.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to read |
| *pin* | : GPIO pin to read |

**Returns**

true if the GPIO pin is high, false if low

**Note**

It is recommended to use the Chip_GPIO_GetPinState() function instead.

Definition at line 121 of file gpio_5411x.h.

### 6.21.2.12 __STATIC_INLINE uint32_t Chip_GPIO_ReadValue ( LPC_GPIO_T ∗ *pGPIO*, uint8_t *portNum* )

Read current bit states for the selected port.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |

| | |
|---:|---|
| *portNum* | : port number to read (supports port 0 only) |

**Returns**

Current value of GPIO port

**Note**

The current states of the bits for the port are read, regardless of whether the GPIO port bits are input or output. It is recommended to use the Chip_GPIO_GetPortValue() function instead.

Definition at line 502 of file gpio_5411x.h.

#### 6.21.2.13 __STATIC_INLINE void Chip_GPIO_SetDir ( LPC_GPIO_T ∗ *pGPIO*, uint8_t *portNum*, uint32_t *bitValue*, uint8_t *out* )

Set Direction for a GPIO port.

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *portNum* | : port Number |
| *bitValue* | : GPIO bit to set |
| *out* | : Direction value, 0 = input, !0 = output |

**Returns**

None

**Note**

Bits set to '0' are not altered. It is recommended to use the Chip_GPIO_SetPortDIR() function instead.

Definition at line 237 of file gpio_5411x.h.

#### 6.21.2.14 __STATIC_INLINE void Chip_GPIO_SetMaskedPortValue ( LPC_GPIO_T ∗ *pGPIO*, uint8_t *port*, uint32_t *value* )

Set all GPIO pin states, but mask via the MASKP0 register.

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *value* | : Value to set all GPIO pin states (0..n) to |

**Returns**

Nothing

Definition at line 363 of file gpio_5411x.h.

#### 6.21.2.15 __STATIC_INLINE void Chip_GPIO_SetPinDIR ( LPC_GPIO_T ∗ *pGPIO*, uint8_t *port*, uint8_t *pin*, bool *output* )

Set GPIO direction for a single GPIO pin.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to set |
| *pin* | : GPIO pin to set direction for |
| *output* | : true for output, false for input |

**Returns**

Nothing

Definition at line 192 of file gpio_5411x.h.

#### 6.21.2.16 __STATIC_INLINE void Chip_GPIO_SetPinDIRInput ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint8_t *pin* )

Set GPIO direction for a single GPIO pin to an input.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to set |
| *pin* | : GPIO pin to set direction on as input |

**Returns**

Nothing

Definition at line 179 of file gpio_5411x.h.

#### 6.21.2.17 __STATIC_INLINE void Chip_GPIO_SetPinDIROutput ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint8_t *pin* )

Set GPIO direction for a single GPIO pin to an output.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to set |
| *pin* | : GPIO pin to set direction on as output |

**Returns**

Nothing

Definition at line 167 of file gpio_5411x.h.

#### 6.21.2.18 __STATIC_INLINE void Chip_GPIO_SetPinOutHigh ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint8_t *pin* )

Set an individual GPIO output pin to the high state.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *pin* | : pin number (0..n) to set high |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output.

Definition at line 417 of file gpio_5411x.h.

#### 6.21.2.19 __STATIC_INLINE void Chip_GPIO_SetPinOutLow ( LPC_GPIO_T ∗ *pGPIO*, uint8_t *port*, uint8_t *pin* )

Set an individual GPIO output pin to the low state.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *pin* | : pin number (0..n) to set low |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output.

Definition at line 460 of file gpio_5411x.h.

#### 6.21.2.20 __STATIC_INLINE void Chip_GPIO_SetPinState ( LPC_GPIO_T ∗ *pGPIO*, uint8_t *port*, uint8_t *pin*, bool *setting* )

Set a GPIO pin state via the GPIO byte register.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to set |
| *pin* | : GPIO pin to set |
| *setting* | : true for high, false for low |

**Returns**

Nothing

**Note**

This function replaces Chip_GPIO_WritePortBit()

Definition at line 108 of file gpio_5411x.h.

#### 6.21.2.21 __STATIC_INLINE void Chip_GPIO_SetPinToggle ( LPC_GPIO_T ∗ *pGPIO*, uint8_t *port*, uint8_t *pin* )

Toggle an individual GPIO output pin to the opposite state.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *pin* | : pin number (0..n) to toggle |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output.

Definition at line 488 of file gpio_5411x.h.

#### 6.21.2.22 __STATIC_INLINE void Chip_GPIO_SetPortDIR ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *pinMask,* bool *outSet* )

Set GPIO direction for a all selected GPIO pins to an input or output.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number |
| *pinMask* | : GPIO pin mask to set direction on (bits 0..b for pins 0..n) |
| *outSet* | : Direction value, false = set as inputs, true = set as outputs |

**Returns**

Nothing

**Note**

Sets multiple GPIO pins to the input direction, each bit's position that is high sets the corresponding pin number for that bit to an input.

Definition at line 285 of file gpio_5411x.h.

#### 6.21.2.23 __STATIC_INLINE void Chip_GPIO_SetPortDIRInput ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *pinMask* )

Set GPIO direction for a all selected GPIO pins to an input.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number |
| *pinMask* | : GPIO pin mask to set direction on as input (bits 0..b for pins 0..n) |

**Returns**

Nothing

**Note**

Sets multiple GPIO pins to the input direction, each bit's position that is high sets the corresponding pin number for that bit to an input.

Definition at line 270 of file gpio_5411x.h.

#### 6.21.2.24 __STATIC_INLINE void Chip_GPIO_SetPortDIROutput ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *pinMask* )

Set GPIO direction for a all selected GPIO pins to an output.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number |
| *pinMask* | : GPIO pin mask to set direction on as output (bits 0..b for pins 0..n) |

**Returns**

Nothing

**Note**

Sets multiple GPIO pins to the output direction, each bit's position that is high sets the corresponding pin number for that bit to an output.

Definition at line 256 of file gpio_5411x.h.

#### 6.21.2.25 __STATIC_INLINE void Chip_GPIO_SetPortMask ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *mask* )

Set GPIO port mask value for GPIO masked read and write.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *mask* | : Mask value for read and write |

**Returns**

Nothing

**Note**

Controls which bits corresponding to PIO0_n are active in the P0MPORT register (bit 0 = PIO0_0, bit 1 = PIO0_1, ..., bit 17 = PIO0_17).

Definition at line 317 of file gpio_5411x.h.

#### 6.21.2.26 __STATIC_INLINE void Chip_GPIO_SetPortOutHigh ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *pins* )

Set selected GPIO output pins to the high state.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *pins* | : pins (0..n) to set high |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output.

Definition at line 403 of file gpio_5411x.h.

#### 6.21.2.27 __STATIC_INLINE void Chip_GPIO_SetPortOutLow ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *pins* )

Set selected GPIO output pins to the low state.

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *pins* | : pins (0..n) to set low |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output.

Definition at line 446 of file gpio_5411x.h.

#### 6.21.2.28 __STATIC_INLINE void Chip_GPIO_SetPortToggle ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *pins* )

Toggle selected GPIO output pins to the opposite state.

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *pins* | : pins (0..n) to toggle |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output.

Definition at line 474 of file gpio_5411x.h.

#### 6.21.2.29 __STATIC_INLINE void Chip_GPIO_SetPortValue ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *port,* uint32_t *value* )

Set all GPIO raw pin states (regardless of masking)

**Parameters**

| | |
|---:|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : port Number (supports port 0 only) |
| *value* | : Value to set all GPIO pin states (0..n) to |

**Returns**

Nothing

Definition at line 340 of file gpio_5411x.h.

#### 6.21.2.30 __STATIC_INLINE void Chip_GPIO_SetValue ( LPC_GPIO_T ∗ *pGPIO,* uint8_t *portNum,* uint32_t *bitValue* )

Set a GPIO port/bit to the high state.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *portNum* | : port number (supports port 0 only) |
| *bitValue* | : bit(s) in the port to set high |

**Returns**

None

**Note**

Any bit set as a '0' will not have it's state changed. This only applies to ports configured as an output. It is recommended to use the Chip_GPIO_SetPortOutHigh() function instead.

Definition at line 389 of file gpio_5411x.h.

#### 6.21.2.31 __STATIC_INLINE void Chip_GPIO_WriteDirBit ( LPC_GPIO_T ∗ *pGPIO,* uint32_t *port,* uint8_t *pin,* bool *setting* )

Set GPIO direction for a single GPIO pin.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to set |
| *pin* | : GPIO pin to set |
| *setting* | : true for output, false for input |

**Returns**

Nothing

**Note**

It is recommended to use the Chip_GPIO_SetPinDIROutput(), Chip_GPIO_SetPinDIRInput() or Chip_GPIO_SetPinDIR() functions instead of this function.

Definition at line 150 of file gpio_5411x.h.

#### 6.21.2.32 __STATIC_INLINE void Chip_GPIO_WritePortBit ( LPC_GPIO_T ∗ *pGPIO,* uint32_t *port,* uint8_t *pin,* bool *setting* )

Set a GPIO port/pin state.

**Parameters**

| | |
|---|---|
| *pGPIO* | : The base of GPIO peripheral on the chip |
| *port* | : GPIO port to set |
| *pin* | : GPIO pin to set |
| *setting* | : true for high, false for low |

**Returns**

Nothing

**Note**

It is recommended to use the Chip_GPIO_SetPinState() function instead.

Definition at line 94 of file gpio_5411x.h.

## 6.22 CHIP: LPC5411X GPIO group driver

### 6.22.1 Detailed Description

### Data Structures

- struct LPC_GPIOGROUPINT_T

  *GPIO grouped interrupt register block structure.*

### Macros

- #define GPIOGR_INT (1 << 0)
- #define GPIOGR_COMB (1 << 1)
- #define GPIOGR_TRIG (1 << 2)

### Functions

- __STATIC_INLINE void Chip_GPIOGP_Init (LPC_GPIOGROUPINT_T ∗pGPIOGPINT)

  *Initialize GPIO group interrupt block.*
- __STATIC_INLINE void Chip_GPIOGP_DeInit (LPC_GPIOGROUPINT_T ∗pGPIOGPINT)

  *De-Initialize GPIO group interrupt block.*
- __STATIC_INLINE void Chip_GPIOGP_ClearIntStatus (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group)

  *Clear interrupt pending status for the selected group.*
- __STATIC_INLINE bool Chip_GPIOGP_GetIntStatus (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group)

  *Returns current GPIO group inetrrupt pending status.*
- __STATIC_INLINE void Chip_GPIOGP_SelectOrMode (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group)

  *Selected GPIO group functionality for trigger on any pin in group (OR mode)*
- __STATIC_INLINE void Chip_GPIOGP_SelectAndMode (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group)

  *Selected GPIO group functionality for trigger on all matching pins in group (AND mode)*
- __STATIC_INLINE void Chip_GPIOGP_SelectEdgeMode (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group)

  *Selected GPIO group functionality edge trigger mode.*
- __STATIC_INLINE void Chip_GPIOGP_SelectLevelMode (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group)

  *Selected GPIO group functionality level trigger mode.*
- __STATIC_INLINE void Chip_GPIOGP_SelectLowLevel (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group, uint8_t port, uint32_t pinMask)

  *Set selected pins for the group and port to low level trigger.*
- __STATIC_INLINE void Chip_GPIOGP_SelectHighLevel (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group, uint8_t port, uint32_t pinMask)

  *Set selected pins for the group and port to high level trigger.*
- __STATIC_INLINE void Chip_GPIOGP_DisableGroupPins (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group, uint8_t port, uint32_t pinMask)

  *Disabled selected pins for the group interrupt.*
- __STATIC_INLINE void Chip_GPIOGP_EnableGroupPins (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group, uint8_t port, uint32_t pinMask)

  *Enable selected pins for the group interrupt.*

## 6.22.2 Macro Definition Documentation

### 6.22.2.1 #define GPIOGR_COMB (1 << 1)

GPIO interrupt OR(0)/AND(1) mode bit

Definition at line 59 of file gpiogroup_5411x.h.

### 6.22.2.2 #define GPIOGR_INT (1 << 0)

LPC5411x GPIO group bit definitionsGPIO interrupt pending/clear bit

Definition at line 58 of file gpiogroup_5411x.h.

### 6.22.2.3 #define GPIOGR_TRIG (1 << 2)

GPIO interrupt edge(0)/level(1) mode bit

Definition at line 60 of file gpiogroup_5411x.h.

## 6.22.3 Function Documentation

### 6.22.3.1 __STATIC_INLINE void Chip_GPIOGP_ClearIntStatus ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group* )

Clear interrupt pending status for the selected group.

**Parameters**

| | |
|---|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |

**Returns**

None

Definition at line 89 of file gpiogroup_5411x.h.

### 6.22.3.2 __STATIC_INLINE void Chip_GPIOGP_DeInit ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT* )

De-Initialize GPIO group interrupt block.

**Parameters**

| | |
|---|---|
| *pGPIOGPINT* | : The base of GPIO group peripheral on the chip |

**Returns**

Nothing

Definition at line 78 of file gpiogroup_5411x.h.

### 6.22.3.3 __STATIC_INLINE void Chip_GPIOGP_DisableGroupPins ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group,* uint8_t *port,* uint32_t *pinMask* )

Disabled selected pins for the group interrupt.

**Parameters**

| | |
|---:|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |
| *port* | : GPIO port number |
| *pinMask* | : Or'ed value of pins to disable interrupt for (bit 0 = pin 0, 1 = pin1, etc.) |

**Returns**

None

**Note**

Disabled pins do not contribute to the group interrupt.

Definition at line 190 of file gpiogroup_5411x.h.

#### 6.22.3.4 __STATIC_INLINE void Chip_GPIOGP_EnableGroupPins ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group,* uint8_t *port,* uint32_t *pinMask* )

Enable selected pins for the group interrupt.

**Parameters**

| | |
|---:|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |
| *port* | : GPIO port number |
| *pinMask* | : Or'ed value of pins to enable interrupt for (bit 0 = pin 0, 1 = pin1, etc.) |

**Returns**

None

**Note**

Enabled pins contribute to the group interrupt.

Definition at line 207 of file gpiogroup_5411x.h.

#### 6.22.3.5 __STATIC_INLINE bool Chip_GPIOGP_GetIntStatus ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group* )

Returns current GPIO group inetrrupt pending status.

**Parameters**

| | |
|---:|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |

**Returns**

true if the group interrupt is pending, otherwise false.

Definition at line 100 of file gpiogroup_5411x.h.

#### 6.22.3.6 __STATIC_INLINE void Chip_GPIOGP_Init ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT* )

Initialize GPIO group interrupt block.

**Parameters**

| | |
|---|---|
| *pGPIOGPINT* | : The base of GPIO group peripheral on the chip |

**Returns**

Nothing

Definition at line 67 of file gpiogroup_5411x.h.

#### 6.22.3.7 __STATIC_INLINE void Chip_GPIOGP_SelectAndMode ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group* )

Selected GPIO group functionality for trigger on all matching pins in group (AND mode)

**Parameters**

| | |
|---|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |

**Returns**

None

Definition at line 122 of file gpiogroup_5411x.h.

#### 6.22.3.8 __STATIC_INLINE void Chip_GPIOGP_SelectEdgeMode ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group* )

Selected GPIO group functionality edge trigger mode.

**Parameters**

| | |
|---|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |

**Returns**

None

Definition at line 133 of file gpiogroup_5411x.h.

#### 6.22.3.9 __STATIC_INLINE void Chip_GPIOGP_SelectHighLevel ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group,* uint8_t *port,* uint32_t *pinMask* )

Set selected pins for the group and port to high level trigger.

**Parameters**

| | |
|---|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |
| *port* | : GPIO port number |
| *pinMask* | : Or'ed value of pins to select for high level (bit 0 = pin 0, 1 = pin1, etc.) |

**Returns**

None

Definition at line 173 of file gpiogroup_5411x.h.

#### 6.22.3.10 __STATIC_INLINE void Chip_GPIOGP_SelectLevelMode ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group* )

Selected GPIO group functionality level trigger mode.

**Parameters**

| | |
|---:|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |

**Returns**

None

Definition at line 144 of file gpiogroup_5411x.h.

### 6.22.3.11 __STATIC_INLINE void Chip_GPIOGP_SelectLowLevel ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group,* uint8_t *port,* uint32_t *pinMask* )

Set selected pins for the group and port to low level trigger.

**Parameters**

| | |
|---:|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |
| *port* | : GPIO port number |
| *pinMask* | : Or'ed value of pins to select for low level (bit 0 = pin 0, 1 = pin1, etc.) |

**Returns**

None

Definition at line 157 of file gpiogroup_5411x.h.

### 6.22.3.12 __STATIC_INLINE void Chip_GPIOGP_SelectOrMode ( LPC_GPIOGROUPINT_T ∗ *pGPIOGPINT,* uint8_t *group* )

Selected GPIO group functionality for trigger on any pin in group (OR mode)

**Parameters**

| | |
|---:|---|
| *pGPIOGPINT* | : Pointer to GPIO group register block |
| *group* | : GPIO group number |

**Returns**

None

Definition at line 111 of file gpiogroup_5411x.h.

## 6.23 CHIP: LPC5411X I2C master-only driver

### 6.23.1 Detailed Description

This driver only works in master mode. To describe the I2C transactions following symbols are used in driver documentation.

### Key to symbols

S (1 bit) : Start bit P (1 bit) : Stop bit Rd/Wr (1 bit) : Read/Write bit. Rd equals 1, Wr equals 0. A, NA (1 bit) : Acknowledge and Not-Acknowledge bit. Addr (7 bits): I2C 7 bit address. Note that this can be expanded as usual to get a 10 bit I2C address. Data (8 bits): A plain data byte. Sometimes, I write DataLow, DataHigh for 16 bit data. [..]: Data sent by I2C device, as opposed to data sent by the host adapter.

### Data Structures

- struct I2CM_XFER_T

  *Master transfer data structure definitions.*

### Macros

- #define I2CM_STATUS_OK 0x00
- #define I2CM_STATUS_ERROR 0x01
- #define I2CM_STATUS_NAK_ADR 0x02
- #define I2CM_STATUS_BUS_ERROR 0x03
- #define I2CM_STATUS_NAK_DAT 0x04
- #define I2CM_STATUS_ARBLOST 0x05
- #define I2CM_STATUS_BUSY 0xFF

### Functions

- static INLINE void Chip_I2CM_SetDutyCycle (LPC_I2C_T ∗pI2C, uint16_t sclH, uint16_t sclL)

  *Sets HIGH and LOW duty cycle registers.*
- void Chip_I2CM_SetBusSpeed (LPC_I2C_T ∗pI2C, uint32_t busSpeed)

  *Set up bus speed for LPC_I2C controller.*
- static INLINE void Chip_I2CM_Enable (LPC_I2C_T ∗pI2C)

  *Enable I2C Master interface.*
- static INLINE void Chip_I2CM_Disable (LPC_I2C_T ∗pI2C)

  *Disable I2C Master interface.*
- static INLINE uint32_t Chip_I2CM_GetStatus (LPC_I2C_T ∗pI2C)

  *Get I2C Status.*
- static INLINE void Chip_I2CM_ClearStatus (LPC_I2C_T ∗pI2C, uint32_t clrStatus)

  *Clear I2C status bits (master)*
- static INLINE bool Chip_I2CM_IsMasterPending (LPC_I2C_T ∗pI2C)

  *Check if I2C Master is pending.*
- static INLINE uint32_t Chip_I2CM_GetMasterState (LPC_I2C_T ∗pI2C)

  *Get current state of the I2C Master.*
- static INLINE void Chip_I2CM_SendStart (LPC_I2C_T ∗pI2C)

  *Transmit START or Repeat-START signal on I2C bus.*
- static INLINE void Chip_I2CM_SendStop (LPC_I2C_T ∗pI2C)

  *Transmit STOP signal on I2C bus.*

- static INLINE void Chip_I2CM_MasterContinue (LPC_I2C_T ∗pI2C)

  *Master Continue transfer operation.*
- static INLINE void Chip_I2CM_WriteByte (LPC_I2C_T ∗pI2C, uint8_t data)

  *Transmit a single data byte through the I2C peripheral (master)*
- static INLINE uint8_t Chip_I2CM_ReadByte (LPC_I2C_T ∗pI2C)

  *Read a single byte data from the I2C peripheral (master)*
- uint32_t Chip_I2CM_XferHandler (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transfer state change handler.*
- void Chip_I2CM_Xfer (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transmit and Receive data in master mode.*
- uint32_t Chip_I2CM_XferBlocking (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transmit and Receive data in master mode.*

## 6.23.2 Macro Definition Documentation

### 6.23.2.1 #define I2CM_STATUS_ARBLOST 0x05

Arbitration lost.

Definition at line 68 of file i2cm_5411x.h.

### 6.23.2.2 #define I2CM_STATUS_BUS_ERROR 0x03

I2C bus error

Definition at line 66 of file i2cm_5411x.h.

### 6.23.2.3 #define I2CM_STATUS_BUSY 0xFF

I2C transmistter is busy.

Definition at line 69 of file i2cm_5411x.h.

### 6.23.2.4 #define I2CM_STATUS_ERROR 0x01

Unknown error condition.

Definition at line 64 of file i2cm_5411x.h.

### 6.23.2.5 #define I2CM_STATUS_NAK_ADR 0x02

No acknowledgement received from slave during address phase.

Definition at line 65 of file i2cm_5411x.h.

### 6.23.2.6 #define I2CM_STATUS_NAK_DAT 0x04

No acknowledgement received from slave during data phase.

Definition at line 67 of file i2cm_5411x.h.

### 6.23.2.7 #define I2CM_STATUS_OK 0x00

I2CM_5411X_STATUS_TYPES I2C master transfer status typesRequested Request was executed successfully.

Definition at line 63 of file i2cm_5411x.h.

## 6.23.3 Function Documentation

### 6.23.3.1 static INLINE void Chip_I2CM_ClearStatus ( LPC_I2C_T ∗ *pI2C*, uint32_t *clrStatus* ) [static]

Clear I2C status bits (master)

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *clrStatus* | : Status bit to clear, ORed Value of I2C_STAT_MSTRARBLOSS and I2C_STAT_MSTST←<br>STPERR |

**Returns**

Nothing

**Note**

This function clears selected status flags.

Definition at line 158 of file i2cm_5411x.h.

### 6.23.3.2 static INLINE void Chip_I2CM_Disable ( LPC_I2C_T ∗ *pI2C* ) [static]

Disable I2C Master interface.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

Definition at line 135 of file i2cm_5411x.h.

### 6.23.3.3 static INLINE void Chip_I2CM_Enable ( LPC_I2C_T ∗ *pI2C* ) [static]

Enable I2C Master interface.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

Definition at line 124 of file i2cm_5411x.h.

### 6.23.3.4 static INLINE uint32_t Chip_I2CM_GetMasterState ( LPC_I2C_T ∗ *pI2C* ) [static]

Get current state of the I2C Master.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Master State Code, a value in the range of 0 - 4

**Note**

After the Master is pending this state code tells the reason for Master pending.

Definition at line 182 of file i2cm_5411x.h.

#### 6.23.3.5 static INLINE uint32_t Chip_I2CM_GetStatus ( LPC_I2C_T ∗ *pI2C* ) [static]

Get I2C Status.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

I2C Status register value

**Note**

This function returns the value of the status register.

Definition at line 146 of file i2cm_5411x.h.

#### 6.23.3.6 static INLINE bool Chip_I2CM_IsMasterPending ( LPC_I2C_T ∗ *pI2C* ) [static]

Check if I2C Master is pending.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Returns TRUE if the Master is pending else returns FALSE

**Note**

Definition at line 170 of file i2cm_5411x.h.

#### 6.23.3.7 static INLINE void Chip_I2CM_MasterContinue ( LPC_I2C_T ∗ *pI2C* ) [static]

Master Continue transfer operation.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function sets the master controller to continue transmission. This should be called only when master is pending. The function writes a complete value to Master Control register, ORing is not advised.

Definition at line 221 of file i2cm_5411x.h.

#### 6.23.3.8 static INLINE uint8_t Chip_I2CM_ReadByte ( LPC_I2C_T ∗ *pI2C* ) [static]

Read a single byte data from the I2C peripheral (master)

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

A single byte of data read

**Note**

This function reads a byte from the I2C receive hold register regardless of I2C state.

Definition at line 247 of file i2cm_5411x.h.

#### 6.23.3.9 static INLINE void Chip_I2CM_SendStart ( LPC_I2C_T ∗ *pI2C* ) [static]

Transmit START or Repeat-START signal on I2C bus.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function sets the controller to transmit START condition when the bus becomes free. This should be called only when master is pending. The function writes a complete value to Master Control register, ORing is not advised.

Definition at line 195 of file i2cm_5411x.h.

#### 6.23.3.10 static INLINE void Chip_I2CM_SendStop ( LPC_I2C_T ∗ *pI2C* ) [static]

Transmit STOP signal on I2C bus.

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function sets the controller to transmit STOP condition. This should be called only when master is pending. The function writes a complete value to Master Control register, ORing is not advised.

Definition at line 208 of file i2cm_5411x.h.

#### 6.23.3.11 void Chip_I2CM_SetBusSpeed ( LPC_I2C_T ∗ *pI2C,* uint32_t *busSpeed* )

Set up bus speed for LPC_I2C controller.

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *busSpeed* | : I2C bus clock rate |

**Returns**

Nothing

**Note**

Per I2C specification the busSpeed should be

- 100000 for Standard mode
- 400000 for Fast mode
- 1000000 for Fast mode plus IOCON registers corresponding to I2C pads should be updated according to the bus mode.  
  This API will also support non-standard bus speeds provided it is an integer multiple of (18, 16, 14, 12, 10, 8, 6, 4).  
  This API will automatically calculate and set divider value.

Definition at line 51 of file i2cm_5411x.c.

#### 6.23.3.12 static INLINE void Chip_I2CM_SetDutyCycle ( LPC_I2C_T ∗ *pI2C,* uint16_t *sclH,* uint16_t *sclL* ) [static]

Sets HIGH and LOW duty cycle registers.

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *sclH* | : Number of I2C_PCLK cycles for the SCL HIGH time value between (2 - 9). |
| *sclL* | : Number of I2C_PCLK cycles for the SCL LOW time value between (2 - 9). |

**Returns**

Nothing

**Note**

The I2C clock divider should be set to the appropriate value before calling this function The I2C baud is determined by the following formula:
I2C_bitFrequency = (I2C_PCLK)/(I2C_CLKDIV ∗ (sclH + sclL))
where I2C_PCLK is the frequency of the System clock and I2C_CLKDIV is I2C clock divider

Definition at line 96 of file i2cm_5411x.h.

### 6.23.3.13 static **INLINE** void Chip_I2CM_WriteByte ( **LPC_I2C_T** ∗ *pI2C,* uint8_t *data* ) [static]

Transmit a single data byte through the I2C peripheral (master)

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *data* | : Byte to transmit |

**Returns**

Nothing

**Note**

This function attempts to place a byte into the I2C Master Data Register

Definition at line 235 of file i2cm_5411x.h.

### 6.23.3.14 void Chip_I2CM_Xfer ( **LPC_I2C_T** ∗ *pI2C,* **I2CM_XFER_T** ∗ *xfer* )

Transmit and Receive data in master mode.

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *xfer* | : Pointer to a I2CM_XFER_T structure see notes below |

**Returns**

Nothing

**Note**

The parameter *xfer* should have its member *slaveAddr* initialized to the 7-Bit slave address to which the master will do the xfer, Bit0 to bit6 should have the address and Bit8 is ignored. During the transfer no code (like event handler) must change the content of the memory pointed to by *xfer*. The member of *xfer*, *txBuff* and *txSz* be initialized to the memory from which the I2C must pick the data to be transfered to slave and the number of bytes to send respectively, similarly *rxBuff* and *rxSz* must have pointer to memroy where data received from slave be stored and the number of data to get from slave respectilvely. Following types of transfers are possible:

- Write-only transfer: When *rxSz* member of *xfer* is set to 0.

  ```
  S Addr Wr [A] txBuff0 [A] txBuff1 [A] ... txBuffN [A] P
  ```

  – If I2CM_XFER_OPTION_IGNORE_NACK is set in *options* memeber
    S Addr Wr [A] txBuff0 [A or NA] ... txBuffN [A or NA] P

- Read-only transfer: When *txSz* member of *xfer* is set to 0.

  ```
  S Addr Rd [A] [rxBuff0] A [rxBuff1] A ... [rxBuffN] NA P
  ```

  - If I2CM_XFER_OPTION_LAST_RX_ACK is set in *options* memeber
    S Addr Rd [A] [rxBuff0] A [rxBuff1] A ... [rxBuffN] A P
- Read-Write transfer: When *rxSz* and @ txSz members of *xfer* are non-zero.

```
S Addr Wr [A] txBuff0 [A] txBuff1 [A] ... txBuffN [A]
    S Addr Rd [A] [rxBuff0] A [rxBuff1] A ... [rxBuffN] NA P
```

Definition at line 163 of file i2cm_5411x.c.

#### 6.23.3.15 uint32_t Chip_I2CM_XferBlocking ( LPC_I2C_T ∗ *pI2C,* I2CM_XFER_T ∗ *xfer* )

Transmit and Receive data in master mode.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *xfer* | : Pointer to a I2CM_XFER_T structure see notes below |

**Returns**

Returns non-zero value on succesful completion of transfer.

**Note**

This function operates same as Chip_I2CM_Xfer(), but is a blocking call.

Definition at line 176 of file i2cm_5411x.c.

#### 6.23.3.16 uint32_t Chip_I2CM_XferHandler ( LPC_I2C_T ∗ *pI2C,* I2CM_XFER_T ∗ *xfer* )

Transfer state change handler.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *xfer* | : Pointer to a I2CM_XFER_T structure see notes below |

**Returns**

Returns non-zero value on completion of transfer. The *status* member of *xfer* structure contains the current status of the transfer at the end of the call.

**Note**

The parameter *xfer* should be same as the one passed to Chip_I2CM_Xfer() routine. This function should be called from the I2C interrupt handler only when a master interrupt occurs.

Definition at line 69 of file i2cm_5411x.c.

## 6.24 CHIP: LPC5411X I2C slave-only driver

### 6.24.1 Detailed Description

This driver only works in slave mode.

### Data Structures

- struct I2CS_XFER_T

### Typedefs

- typedef void(∗ I2CSlaveXferStart) (uint8_t addr)

  *I2C slave service start callback This callback is called from the I2C slave handler when an I2C slave address is received and needs servicing. It's used to indicate the start of a slave transfer that will happen on the slave bus.*
- typedef uint8_t(∗ I2CSlaveXferSend) (uint8_t ∗data)

  *I2C slave send data callback This callback is called from the I2C slave handler when an I2C slave address needs data to send.*
  *If you want to NAK the master, return I2C_SLVCTL_SLVNACK to the caller. Return I2C_SLVCTL_SLVCONTINUE or 0 to the caller for normal non-DMA data transfer. If you've setup a DMA descriptor for the transfer, return I2C_SLVCTL_SLVDMA to the caller.*
- typedef uint8_t(∗ I2CSlaveXferRecv) (uint8_t data)

  *I2C slave receive data callback This callback is called from the I2C slave handler when an I2C slave address has receive data.*
  *If you want to NAK the master, return I2C_SLVCTL_SLVNACK to the caller. Return I2C_SLVCTL_SLVCONTINUE or 0 to the caller for normal non-DMA data transfer. If you've setup a DMA descriptor for the transfer, return I2C_SLVCTL_SLVDMA to the caller.*
- typedef void(∗ I2CSlaveXferDone) (void)

  *I2C slave service done callback This callback is called from the I2C slave handler when an I2C slave transfer is completed. It's used to indicate the end of a slave transfer.*

### Functions

- __STATIC_INLINE void Chip_I2CS_Enable (LPC_I2C_T ∗pI2C)

  *Enable I2C slave interface.*
- __STATIC_INLINE void Chip_I2CS_Disable (LPC_I2C_T ∗pI2C)

  *Disable I2C slave interface.*
- __STATIC_INLINE uint32_t Chip_I2CS_GetStatus (LPC_I2C_T ∗pI2C)

  *Get I2C Status.*
- __STATIC_INLINE void Chip_I2CS_ClearStatus (LPC_I2C_T ∗pI2C, uint32_t clrStatus)

  *Clear I2C status bits (slave)*
- __STATIC_INLINE bool Chip_I2CS_IsSlavePending (LPC_I2C_T ∗pI2C)

  *Check if I2C slave is pending.*
- __STATIC_INLINE bool Chip_I2CS_IsSlaveSelected (LPC_I2C_T ∗pI2C)

  *Check if I2C slave is selected.*
- __STATIC_INLINE bool Chip_I2CS_IsSlaveDeSelected (LPC_I2C_T ∗pI2C)

  *Check if I2C slave is deselected.*
- __STATIC_INLINE uint32_t Chip_I2CS_GetSlaveState (LPC_I2C_T ∗pI2C)

  *Get current state of the I2C slave.*
- __STATIC_INLINE uint32_t Chip_I2CS_GetSlaveMatchIndex (LPC_I2C_T ∗pI2C)

*Returns the current slave address match index.*

- __STATIC_INLINE void Chip_I2CS_SlaveContinue (LPC_I2C_T ∗pI2C)

  *Slave Continue transfer operation (ACK)*
- __STATIC_INLINE void Chip_I2CS_SlaveNACK (LPC_I2C_T ∗pI2C)

  *Slave NACK operation.*
- __STATIC_INLINE void Chip_I2CS_SlaveEnableDMA (LPC_I2C_T ∗pI2C)

  *Enable slave DMA operation.*
- __STATIC_INLINE void Chip_I2CS_SlaveDisableDMA (LPC_I2C_T ∗pI2C)

  *Disable slave DMA operation.*
- __STATIC_INLINE void Chip_I2CS_WriteByte (LPC_I2C_T ∗pI2C, uint8_t data)

  *Transmit a single data byte through the I2C peripheral (slave)*
- __STATIC_INLINE uint8_t Chip_I2CS_ReadByte (LPC_I2C_T ∗pI2C)

  *Read a single byte data from the I2C peripheral (slave)*
- __STATIC_INLINE void Chip_I2CS_SetSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum, uint8_t slvAddr)

  *Set a I2C slave address for slave operation.*
- __STATIC_INLINE uint8_t Chip_I2CS_GetSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum)

  *Return a I2C programmed slave address.*
- __STATIC_INLINE void Chip_I2CS_EnableSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum)

  *Enable a I2C address.*
- __STATIC_INLINE void Chip_I2CS_DisableSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum)

  *Disable a I2C address.*
- __STATIC_INLINE void Chip_I2CS_SetSlaveQual0 (LPC_I2C_T ∗pI2C, bool extend, uint8_t slvAddr)

  *Setup slave qialifier address.*
- uint32_t Chip_I2CS_XferHandler (LPC_I2C_T ∗pI2C, const I2CS_XFER_T ∗xfers)

  *Slave transfer state change handler.*

## 6.24.2 Typedef Documentation

### 6.24.2.1 typedef void(∗ I2CSlaveXferDone) (void)

I2C slave service done callback This callback is called from the I2C slave handler when an I2C slave transfer is completed. It's used to indicate the end of a slave transfer.

Definition at line 76 of file i2cs_5411x.h.

### 6.24.2.2 typedef uint8_t(∗ I2CSlaveXferRecv) (uint8_t data)

I2C slave receive data callback This callback is called from the I2C slave handler when an I2C slave address has receive data.
If you want to NAK the master, return I2C_SLVCTL_SLVNACK to the caller. Return I2C_SLVCTL_SLVCONTINUE or 0 to the caller for normal non-DMA data transfer. If you've setup a DMA descriptor for the transfer, return I2C_SLVCTL_SLVDMA to the caller.

Definition at line 70 of file i2cs_5411x.h.

### 6.24.2.3 typedef uint8_t(∗ I2CSlaveXferSend) (uint8_t ∗data)

I2C slave send data callback This callback is called from the I2C slave handler when an I2C slave address needs data to send.
If you want to NAK the master, return I2C_SLVCTL_SLVNACK to the caller. Return I2C_SLVCTL_SLVCONTINUE or 0 to the caller for normal non-DMA data transfer. If you've setup a DMA descriptor for the transfer, return

I2C_SLVCTL_SLVDMA to the caller.

Definition at line 61 of file i2cs_5411x.h.

#### 6.24.2.4 typedef void(∗ I2CSlaveXferStart) (uint8_t addr)

I2C slave service start callback This callback is called from the I2C slave handler when an I2C slave address is received and needs servicing. It's used to indicate the start of a slave transfer that will happen on the slave bus.

Definition at line 52 of file i2cs_5411x.h.

### 6.24.3 Function Documentation

#### 6.24.3.1 __STATIC_INLINE void Chip_I2CS_ClearStatus ( LPC_I2C_T ∗ *pI2C,* uint32_t *clrStatus* )

Clear I2C status bits (slave)

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *clrStatus* | : Status bit to clear, must be I2C_STAT_SLVDESEL |

**Returns**

Nothing

**Note**

This function clears selected status flags.

Definition at line 149 of file i2cs_5411x.h.

#### 6.24.3.2 __STATIC_INLINE void Chip_I2CS_Disable ( LPC_I2C_T ∗ *pI2C* )

Disable I2C slave interface.

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

Definition at line 126 of file i2cs_5411x.h.

#### 6.24.3.3 __STATIC_INLINE void Chip_I2CS_DisableSlaveAddr ( LPC_I2C_T ∗ *pI2C,* uint8_t *slvNum* )

Disable a I2C address.

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |

| | |
|---|---|
| *slvNum* | : Possible slave address number, between 0 - 3 |

**Returns**

Nothing

Definition at line 326 of file i2cs_5411x.h.

#### 6.24.3.4 __STATIC_INLINE void Chip_I2CS_Enable ( LPC_I2C_T ∗ *pI2C* )

Enable I2C slave interface.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

Do not call this function until the slave interface is fully configured.

Definition at line 116 of file i2cs_5411x.h.

#### 6.24.3.5 __STATIC_INLINE void Chip_I2CS_EnableSlaveAddr ( LPC_I2C_T ∗ *pI2C,* uint8_t *slvNum* )

Enable a I2C address.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *slvNum* | : Possible slave address number, between 0 - 3 |

**Returns**

Nothing

Definition at line 315 of file i2cs_5411x.h.

#### 6.24.3.6 __STATIC_INLINE uint8_t Chip_I2CS_GetSlaveAddr ( LPC_I2C_T ∗ *pI2C,* uint8_t *slvNum* )

Return a I2C programmed slave address.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *slvNum* | : Possible slave address number, between 0 - 3 |

**Returns**

Nothing

Definition at line 304 of file i2cs_5411x.h.

#### 6.24.3.7 __STATIC_INLINE uint32_t Chip_I2CS_GetSlaveMatchIndex ( LPC_I2C_T ∗ *pI2C* )

Returns the current slave address match index.

**Parameters**

|  |  |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

slave match index, 0 - 3

Definition at line 204 of file i2cs_5411x.h.

#### 6.24.3.8 __STATIC_INLINE uint32_t Chip_I2CS_GetSlaveState ( LPC_I2C_T ∗ *pI2C* )

Get current state of the I2C slave.

**Parameters**

|  |  |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

slave State Code, a value of type I2C_STAT_SLVCODE_∗

**Note**

After the slave is pending this state code tells the reason for slave pending.

Definition at line 194 of file i2cs_5411x.h.

#### 6.24.3.9 __STATIC_INLINE uint32_t Chip_I2CS_GetStatus ( LPC_I2C_T ∗ *pI2C* )

Get I2C Status.

**Parameters**

|  |  |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

I2C Status register value

**Note**

This function returns the value of the status register.

Definition at line 137 of file i2cs_5411x.h.

#### 6.24.3.10 __STATIC_INLINE bool Chip_I2CS_IsSlaveDeSelected ( LPC_I2C_T ∗ *pI2C* )

Check if I2C slave is deselected.

**Parameters**

|  |  |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Returns TRUE if the slave is is deselected, otherwise FALSE

**Note**

Definition at line 182 of file i2cs_5411x.h.

#### 6.24.3.11 __STATIC_INLINE bool Chip_I2CS_IsSlavePending ( LPC_I2C_T ∗ *pI2C* )

Check if I2C slave is pending.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Returns TRUE if the slave is pending else returns FALSE

**Note**

Definition at line 160 of file i2cs_5411x.h.

#### 6.24.3.12 __STATIC_INLINE bool Chip_I2CS_IsSlaveSelected ( LPC_I2C_T ∗ *pI2C* )

Check if I2C slave is selected.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Returns TRUE if the slave is is selected, otherwise FALSE

**Note**

Definition at line 171 of file i2cs_5411x.h.

#### 6.24.3.13 __STATIC_INLINE uint8_t Chip_I2CS_ReadByte ( LPC_I2C_T ∗ *pI2C* )

Read a single byte data from the I2C peripheral (slave)

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

A single byte of data read

**Note**

This function reads a byte from the I2C receive hold register regardless of I2C state.

Definition at line 279 of file i2cs_5411x.h.

#### 6.24.3.14 __STATIC_INLINE void Chip_I2CS_SetSlaveAddr ( LPC_I2C_T ∗ *pI2C,* uint8_t *slvNum,* uint8_t *slvAddr* )

Set a I2C slave address for slave operation.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *slvNum* | : Possible slave address number, between 0 - 3 |
| *slvAddr* | : Slave Address for the index (7-bits, bit 7 = 0) |

**Returns**

Nothing

**Note**

Setting a slave address also enables the slave address. Do not 'pre-shift' the slave address.

Definition at line 293 of file i2cs_5411x.h.

#### 6.24.3.15 __STATIC_INLINE void Chip_I2CS_SetSlaveQual0 ( LPC_I2C_T ∗ *pI2C,* bool *extend,* uint8_t *slvAddr* )

Setup slave qialifier address.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *extend* | : true to extend I2C slave detect address 0 range, or false to match to corresponding bits |
| *slvAddr* | : Slave address qualifier, see SLVQUAL0 register in User Manual |

**Returns**

Nothing

**Note**

Do not 'pre-shift' the slave address.

Definition at line 339 of file i2cs_5411x.h.

#### 6.24.3.16 __STATIC_INLINE void Chip_I2CS_SlaveContinue ( LPC_I2C_T ∗ *pI2C* )

Slave Continue transfer operation (ACK)

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function sets the slave controller to continue transmission. This should be called only when slave is pending. The function writes a complete value to slave Control register, ORing is not advised.

Definition at line 217 of file i2cs_5411x.h.

#### 6.24.3.17 __STATIC_INLINE void Chip_I2CS_SlaveDisableDMA ( LPC_I2C_T ∗ *pI2C* )

Disable slave DMA operation.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function disables DMA mode for the slave controller.

Definition at line 253 of file i2cs_5411x.h.

#### 6.24.3.18 __STATIC_INLINE void Chip_I2CS_SlaveEnableDMA ( LPC_I2C_T ∗ *pI2C* )

Enable slave DMA operation.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function enables DMA mode for the slave controller. In DMA mode, the 'continue' and 'NACK' operations aren't used and the I2C slave controller will automatically NACK any bytes beyond the available DMA buffer size.

Definition at line 242 of file i2cs_5411x.h.

#### 6.24.3.19 __STATIC_INLINE void Chip_I2CS_SlaveNACK ( LPC_I2C_T ∗ *pI2C* )

Slave NACK operation.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function sets the slave controller to NAK the master.

Definition at line 228 of file i2cs_5411x.h.

#### 6.24.3.20 __STATIC_INLINE void Chip_I2CS_WriteByte ( LPC_I2C_T ∗ *pI2C,* uint8_t *data* )

Transmit a single data byte through the I2C peripheral (slave)

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *data* | : Byte to transmit |

**Returns**

Nothing

**Note**

This function attempts to place a byte into the I2C slave Data Register

Definition at line 267 of file i2cs_5411x.h.

#### 6.24.3.21 uint32_t Chip_I2CS_XferHandler ( LPC_I2C_T ∗ *pI2C,* const I2CS_XFER_T ∗ *xfers* )

Slave transfer state change handler.

**Parameters**

| | |
|---:|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *xfers* | : Pointer to a I2CS_MULTI_XFER_T structure see notes below |

**Returns**

Returns non-zero value on completion of transfer or NAK

**Note**

See I2CS_XFER_T for more information on this function. When using this function, the I2C_INTENSET_↩ SLVPENDING and I2C_INTENSET_SLVDESEL interrupts should be enabled and setup in the I2C interrupt handler to call this function when they fire.

Definition at line 51 of file i2cs_5411x.c.

# 6.25 CHIP: LPC5411X IOCON register block and driver

## 6.25.1 Detailed Description

### Data Structures

- struct LPC_IOCON_T

  *LPC5411X IO Configuration Unit register block structure.*
- struct PINMUX_GRP_T

  *Array of IOCON pin definitions passed to Chip_IOCON_SetPinMuxing() must be in this format.*

### Macros

- #define IOCON_FUNC0 0x0
- #define IOCON_FUNC1 0x1
- #define IOCON_FUNC2 0x2
- #define IOCON_FUNC3 0x3
- #define IOCON_FUNC4 0x4
- #define IOCON_FUNC5 0x5
- #define IOCON_FUNC6 0x6
- #define IOCON_FUNC7 0x7
- #define IOCON_MODE_INACT (0x0 << 3)
- #define IOCON_MODE_PULLDOWN (0x1 << 3)
- #define IOCON_MODE_PULLUP (0x2 << 3)
- #define IOCON_MODE_REPEATER (0x3 << 3)
- #define IOCON_HYS_EN (0x1 << 5)
- #define IOCON_GPIO_MODE (0x1 << 5)
- #define IOCON_I2C_SLEW (0x1 << 5)
- #define IOCON_INV_EN (0x1 << 6)
- #define IOCON_ANALOG_EN (0x0 << 7)
- #define IOCON_DIGITAL_EN (0x1 << 7)
- #define IOCON_STDI2C_EN (0x1 << 8)
- #define IOCON_FASTI2C_EN (0x3 << 8)
- #define IOCON_INPFILT_OFF (0x1 << 8)
- #define IOCON_INPFILT_ON (0x0 << 8)
- #define IOCON_OPENDRAIN_EN (0x1 << 10)
- #define IOCON_S_MODE_0CLK (0x0 << 11)
- #define IOCON_S_MODE_1CLK (0x1 << 11)
- #define IOCON_S_MODE_2CLK (0x2 << 11)
- #define IOCON_S_MODE_3CLK (0x3 << 11)
- #define IOCON_S_MODE(clks) ((clks) << 11)
- #define IOCON_CLKDIV(div) ((div) << 13)

### Functions

- __STATIC_INLINE void Chip_IOCON_PinMuxSet (LPC_IOCON_T ∗pIOCON, uint8_t port, uint8_t pin, uint32_t modefunc)

  *Sets I/O Control pin mux.*
- __STATIC_INLINE void Chip_IOCON_PinMux (LPC_IOCON_T ∗pIOCON, uint8_t port, uint8_t pin, uint16_t mode, uint8_t func)

  *I/O Control pin mux.*
- __STATIC_INLINE void Chip_IOCON_SetPinMuxing (LPC_IOCON_T ∗pIOCON, const PINMUX_GRP_T ∗pinArray, uint32_t arrayLength)

  *Set all I/O Control pin muxing.*

## 6.25.2 Macro Definition Documentation

### 6.25.2.1 #define IOCON_ANALOG_EN (0x0 << 7)

Enables analog function by setting 0 to bit 7

Definition at line 81 of file iocon_5411x.h.

### 6.25.2.2 #define IOCON_CLKDIV( *div* ) ((div) << 13)

Select peripheral clock divider for input filter sampling clock, 2^n, n=0-6

Definition at line 93 of file iocon_5411x.h.

### 6.25.2.3 #define IOCON_DIGITAL_EN (0x1 << 7)

Enables digital function by setting 1 to bit 7(default)

Definition at line 82 of file iocon_5411x.h.

### 6.25.2.4 #define IOCON_FASTI2C_EN (0x3 << 8)

I2C Fast-mode Plus and high-speed slave

Definition at line 84 of file iocon_5411x.h.

### 6.25.2.5 #define IOCON_FUNC0 0x0

IOCON function and mode selection definitions See the User Manual for specific modes and functions supported by the various LPC15XX pins.Selects pin function 0

Definition at line 65 of file iocon_5411x.h.

### 6.25.2.6 #define IOCON_FUNC1 0x1

Selects pin function 1

Definition at line 66 of file iocon_5411x.h.

### 6.25.2.7 #define IOCON_FUNC2 0x2

Selects pin function 2

Definition at line 67 of file iocon_5411x.h.

### 6.25.2.8 #define IOCON_FUNC3 0x3

Selects pin function 3

Definition at line 68 of file iocon_5411x.h.

### 6.25.2.9 #define IOCON_FUNC4 0x4

Selects pin function 4

Definition at line 69 of file iocon_5411x.h.

#### 6.25.2.10 #define IOCON_FUNC5 0x5

Selects pin function 5

Definition at line 70 of file iocon_5411x.h.

#### 6.25.2.11 #define IOCON_FUNC6 0x6

Selects pin function 6

Definition at line 71 of file iocon_5411x.h.

#### 6.25.2.12 #define IOCON_FUNC7 0x7

Selects pin function 7

Definition at line 72 of file iocon_5411x.h.

#### 6.25.2.13 #define IOCON_GPIO_MODE (0x1 << 5)

GPIO Mode

Definition at line 78 of file iocon_5411x.h.

#### 6.25.2.14 #define IOCON_HYS_EN (0x1 << 5)

Enables hysteresis

Definition at line 77 of file iocon_5411x.h.

#### 6.25.2.15 #define IOCON_I2C_SLEW (0x1 << 5)

I2C Slew Rate Control

Definition at line 79 of file iocon_5411x.h.

#### 6.25.2.16 #define IOCON_INPFILT_OFF (0x1 << 8)

Input filter Off for GPIO pins

Definition at line 85 of file iocon_5411x.h.

#### 6.25.2.17 #define IOCON_INPFILT_ON (0x0 << 8)

Input filter On for GPIO pins

Definition at line 86 of file iocon_5411x.h.

#### 6.25.2.18 #define IOCON_INV_EN (0x1 << 6)

Enables invert function on input

Definition at line 80 of file iocon_5411x.h.

#### 6.25.2.19 #define IOCON_MODE_INACT (0x0 << 3)

No addition pin function

Definition at line 73 of file iocon_5411x.h.

#### 6.25.2.20 #define IOCON_MODE_PULLDOWN (0x1 << 3)

Selects pull-down function

Definition at line 74 of file iocon_5411x.h.

#### 6.25.2.21 #define IOCON_MODE_PULLUP (0x2 << 3)

Selects pull-up function

Definition at line 75 of file iocon_5411x.h.

#### 6.25.2.22 #define IOCON_MODE_REPEATER (0x3 << 3)

Selects pin repeater function

Definition at line 76 of file iocon_5411x.h.

#### 6.25.2.23 #define IOCON_OPENDRAIN_EN (0x1 << 10)

Enables open-drain function

Definition at line 87 of file iocon_5411x.h.

#### 6.25.2.24 #define IOCON_S_MODE( *clks* ) ((clks) << 11)

Select clocks for digital input filter mode

Definition at line 92 of file iocon_5411x.h.

#### 6.25.2.25 #define IOCON_S_MODE_0CLK (0x0 << 11)

Bypass input filter

Definition at line 88 of file iocon_5411x.h.

#### 6.25.2.26 #define IOCON_S_MODE_1CLK (0x1 << 11)

Input pulses shorter than 1 filter clock are rejected

Definition at line 89 of file iocon_5411x.h.

#### 6.25.2.27 #define IOCON_S_MODE_2CLK (0x2 << 11)

Input pulses shorter than 2 filter clock2 are rejected

Definition at line 90 of file iocon_5411x.h.

#### 6.25.2.28 #define IOCON_S_MODE_3CLK (0x3 << 11)

Input pulses shorter than 3 filter clock2 are rejected

Definition at line 91 of file iocon_5411x.h.

#### 6.25.2.29 #define IOCON_STDI2C_EN (0x1 << 8)

I2C standard mode/fast-mode

Definition at line 83 of file iocon_5411x.h.

### 6.25.3 Function Documentation

#### 6.25.3.1 __STATIC_INLINE void Chip_IOCON_PinMux ( LPC_IOCON_T ∗ *pIOCON,* uint8_t *port,* uint8_t *pin,* uint16_t *mode,* uint8_t *func* )

I/O Control pin mux.

**Parameters**

| | |
|---|---|
| *pIOCON* | : The base of IOCON peripheral on the chip |
| *port* | : GPIO port to mux |
| *pin* | : GPIO pin to mux |
| *mode* | : OR'ed values or type IOCON_∗ |
| *func* | : Pin function, value of type IOCON_FUNC? |

**Returns**

Nothing

Definition at line 117 of file iocon_5411x.h.

#### 6.25.3.2 __STATIC_INLINE void Chip_IOCON_PinMuxSet ( LPC_IOCON_T ∗ *pIOCON,* uint8_t *port,* uint8_t *pin,* uint32_t *modefunc* )

Sets I/O Control pin mux.

**Parameters**

| | |
|---|---|
| *pIOCON* | : The base of IOCON peripheral on the chip |
| *port* | : GPIO port to mux |
| *pin* | : GPIO pin to mux |
| *modefunc* | : OR'ed values or type IOCON_∗ |

**Returns**

Nothing

Definition at line 103 of file iocon_5411x.h.

#### 6.25.3.3 __STATIC_INLINE void Chip_IOCON_SetPinMuxing ( LPC_IOCON_T ∗ *pIOCON,* const PINMUX_GRP_T ∗ *pinArray,* uint32_t *arrayLength* )

Set all I/O Control pin muxing.

### Parameters

| | |
|---|---|
| *pIOCON* | : The base of IOCON peripheral on the chip |
| *pinArray* | : Pointer to array of pin mux selections |
| *arrayLength* | : Number of entries in pinArray |

### Returns

Nothing

Definition at line 129 of file iocon_5411x.h.

# 6.26 CHIP: LPC5411X Input Mux Registers and Driver

## 6.26.1 Detailed Description

### Data Structures

- struct LPC_INMUX_T

  *LPC5411X Input Mux Register Block Structure.*

### Enumerations

- enum DMA_TRIGSRC_T {
  DMATRIG_ADC0_SEQA_IRQ = 0, DMATRIG_ADC0_SEQB_IRQ, DMATRIG_SCT0_DMA0, DMATRIG_↩
  SCT0_DMA1,
  DMATRIG_TIMER0_MATCH0, DMATRIG_TIMER0_MATCH1, DMATRIG_TIMER1_MATCH0, DMATRIG↩
  _TIMER2_MATCH0,
  DMATRIG_TIMER2_MATCH1, DMATRIG_TIMER3_MATCH0, DMATRIG_TIMER4_MATCH0, DMATRIG↩
  _TIMER4_MATCH1,
  DMATRIG_PININT0, DMATRIG_PININT1, DMATRIG_PININT2, DMATRIG_PININT3,
  DMATRIG_OUTMUX0, DMATRIG_OUTMUX1, DMATRIG_OUTMUX2, DMATRIG_OUTMUX3 }
- enum FREQMSR_SRC_T {
  FREQMSR_CLKIN = 0, FREQMSR_FRO12MHZ, FREQMSR_WDOSC, FREQMSR_32KHZOSC,
  FREQ_MEAS_MAIN_CLK, FREQMSR_PIO0_4, FREQMSR_PIO0_20, FREQMSR_PIO0_24,
  FREQMSR_PIO1_4 }

### Functions

- __STATIC_INLINE void Chip_INMUX_PinIntSel (uint8_t pintSel, uint8_t portNum, uint8_t pinNum)

  *GPIO Pin Interrupt Pin Select (sets PINTSEL register)*
- __STATIC_INLINE void Chip_INMUX_SetDMATrigger (uint8_t ch, DMA_TRIGSRC_T trig)

  *Select a trigger source for a DMA channel.*
- __STATIC_INLINE void Chip_INMUX_SetDMAOutMux (uint8_t index, uint8_t dmaCh)

  *Selects a DMA trigger source for the DMATRIG_OUTMUXn IDs.*
- __STATIC_INLINE void Chip_INMUX_SetFreqMeasRefClock (FREQMSR_SRC_T ref)

  *Selects a reference clock used with the frequency measure function.*
- __STATIC_INLINE void Chip_INMUX_SetFreqMeasTargClock (FREQMSR_SRC_T targ)

  *Selects a target clock used with the frequency measure function.*

## 6.26.2 Enumeration Type Documentation

### 6.26.2.1 enum DMA_TRIGSRC_T

Enumerator

| | |
|---|---|
| ***DMATRIG_ADC0_SEQA_IRQ*** | ADC0 sequencer A interrupt as trigger |
| ***DMATRIG_ADC0_SEQB_IRQ*** | ADC0 sequencer B interrupt as trigger |
| ***DMATRIG_SCT0_DMA0*** | SCT 0, DMA 0 as trigger |
| ***DMATRIG_SCT0_DMA1*** | SCT 1, DMA 1 as trigger |
| ***DMATRIG_TIMER0_MATCH0*** | Timer 0, match 0 trigger |
| ***DMATRIG_TIMER0_MATCH1*** | Timer 0, match 1 trigger |
| ***DMATRIG_TIMER1_MATCH0*** | Timer 1, match 0 trigger |
| ***DMATRIG_TIMER2_MATCH0*** | Timer 2, match 0 trigger |

***DMATRIG_TIMER2_MATCH1*** Timer 2, match 1 trigger

***DMATRIG_TIMER3_MATCH0*** Timer 3, match 0 trigger

***DMATRIG_TIMER4_MATCH0*** Timer 4, match 0 trigger

***DMATRIG_TIMER4_MATCH1*** Timer 4, match 1 trigger

***DMATRIG_PININT0*** Pin interrupt 0 trigger

***DMATRIG_PININT1*** Pin interrupt 1 trigger

***DMATRIG_PININT2*** Pin interrupt 2 trigger

***DMATRIG_PININT3*** Pin interrupt 3 trigger

***DMATRIG_OUTMUX0*** DMA trigger tied to this source, Select with Chip_INMUX_SetDMAOutMux

***DMATRIG_OUTMUX1*** DMA trigger tied to this source, Select with Chip_INMUX_SetDMAOutMux

***DMATRIG_OUTMUX2*** DMA trigger tied to this source, Select with Chip_INMUX_SetDMAOutMux

***DMATRIG_OUTMUX3*** DMA trigger tied to this source, Select with Chip_INMUX_SetDMAOutMux

Definition at line 71 of file inmux_5411x.h.

#### 6.26.2.2 enum FREQMSR_SRC_T

**Enumerator**

***FREQMSR_CLKIN*** CLKIN pin

***FREQMSR_FRO12MHZ*** FRO 12-MHz

***FREQMSR_WDOSC*** Watchdog oscillator

***FREQMSR_32KHZOSC*** 32KHz (RTC) oscillator rate

***FREQ_MEAS_MAIN_CLK*** main system clock

***FREQMSR_PIO0_4*** External pin PIO0_4 as input rate

***FREQMSR_PIO0_20*** External pin PIO0_20 as input rate

***FREQMSR_PIO0_24*** External pin PIO0_24 as input rate

***FREQMSR_PIO1_4*** External pin PIO1_4 as input rate

Definition at line 119 of file inmux_5411x.h.

### 6.26.3 Function Documentation

#### 6.26.3.1 __STATIC_INLINE void Chip_INMUX_PinIntSel ( uint8_t *pintSel,* uint8_t *portNum,* uint8_t *pinNum* )

GPIO Pin Interrupt Pin Select (sets PINTSEL register)

**Parameters**

| | |
|---|---|
| *pintSel* | : GPIO PINTSEL interrupt, should be: 0 to 7 |
| *portNum* | : GPIO port number interrupt, should be: 0 to 1 |
| *pinNum* | : GPIO pin number Interrupt, should be: 0 to 31 |

**Returns**

Nothing

Definition at line 65 of file inmux_5411x.h.

#### 6.26.3.2 __STATIC_INLINE void Chip_INMUX_SetDMAOutMux ( uint8_t *index,* uint8_t *dmaCh* )

Selects a DMA trigger source for the DMATRIG_OUTMUXn IDs.

**Parameters**

| | |
|---|---|
| *index* | : Select 0 to 3 to sets the source for DMATRIG_OUTMUX0 to DMATRIG_OUTMUX3 |
| *dmaCh* | : DMA channel to select for DMATRIG_OUTMUXn source |

**Returns**

Nothing

**Note**

This function sets the DMA trigger (out) source used with the DMATRIG_OUTMUXn trigger source.

Definition at line 113 of file inmux_5411x.h.

#### 6.26.3.3 __STATIC_INLINE void Chip_INMUX_SetDMATrigger ( uint8_t *ch*, DMA_TRIGSRC_T *trig* )

Select a trigger source for a DMA channel.

**Parameters**

| | |
|---|---|
| *ch* | : DMA channel number |
| *trig* | : Trigger source for the DMA channel |

**Returns**

Nothing

Definition at line 100 of file inmux_5411x.h.

#### 6.26.3.4 __STATIC_INLINE void Chip_INMUX_SetFreqMeasRefClock ( FREQMSR_SRC_T *ref* )

Selects a reference clock used with the frequency measure function.

**Parameters**

| | |
|---|---|
| *ref* | : Frequency measure function reference clock |

**Returns**

Nothing

Definition at line 136 of file inmux_5411x.h.

#### 6.26.3.5 __STATIC_INLINE void Chip_INMUX_SetFreqMeasTargClock ( FREQMSR_SRC_T *targ* )

Selects a target clock used with the frequency measure function.

**Parameters**

| | |
|---|---|
| *targ* | : Frequency measure function reference clock |

**Returns**

Nothing

Definition at line 146 of file inmux_5411x.h.

## 6.27 CHIP: LPC5411X M0 core CMSIS include file

### 6.27.1 Detailed Description

### Modules

- CHIP: LPC5411X M0 core Cortex CMSIS definitions
- CHIP_5411X: LPC5411X M0 core peripheral interrupt numbers

# 6.28 CHIP: LPC5411X M0 core Cortex CMSIS definitions

## 6.28.1 Detailed Description

### Macros

- #define __CM0PLUS_REV 0x0001
- #define __MPU_PRESENT 0
- #define __NVIC_PRIO_BITS 2
- #define __Vendor_SysTickConfig 0
- #define __VTOR_PRESENT 1
- #define TIMER0_IRQn CT32B0_IRQn

  *interrupt Alias*
- #define TIMER1_IRQn CT32B1_IRQn
- #define TIMER2_IRQn CT32B2_IRQn
- #define TIMER3_IRQn CT32B3_IRQn
- #define TIMER4_IRQn CT32B4_IRQn
- #define SCT_IRQn SCT0_IRQn
- #define ADC0_SEQA_IRQn ADC_SEQA_IRQn
- #define ADC0_SEQB_IRQn ADC_SEQB_IRQn
- #define ADC0_THCMP_IRQn ADC_THCMP_IRQn
- #define TIMER0_IRQHandler CT32B0_IRQHandler

  *Interrupt handler Alias.*
- #define TIMER1_IRQHandler CT32B1_IRQHandler
- #define TIMER2_IRQHandler CT32B2_IRQHandler
- #define TIMER3_IRQHandler CT32B3_IRQHandler
- #define TIMER4_IRQHandler CT32B4_IRQHandler
- #define SCT_IRQHandler SCT0_IRQHandler
- #define ADC0_SEQA_IRQHandler ADC_SEQA_IRQHandler
- #define ADC0_SEQB_IRQHandler ADC_SEQB_IRQHandler
- #define ADC0_THCMP_IRQHandler ADC_THCMP_IRQHandler

## 6.28.2 Macro Definition Documentation

### 6.28.2.1 #define __CM0PLUS_REV 0x0001

Cortex-M0PLUS Core Revision

Definition at line 136 of file cmsis_5411x_m0.h.

### 6.28.2.2 #define __MPU_PRESENT 0

MPU present or not

Definition at line 137 of file cmsis_5411x_m0.h.

### 6.28.2.3 #define __NVIC_PRIO_BITS 2

Number of Bits used for Priority Levels

Definition at line 138 of file cmsis_5411x_m0.h.

#### 6.28.2.4 #define __Vendor_SysTickConfig 0

Set to 1 if different SysTick Config is used

Definition at line 139 of file cmsis_5411x_m0.h.

#### 6.28.2.5 #define __VTOR_PRESENT 1

Definition at line 140 of file cmsis_5411x_m0.h.

#### 6.28.2.6 #define ADC0_SEQA_IRQHandler ADC_SEQA_IRQHandler

Definition at line 160 of file cmsis_5411x_m0.h.

#### 6.28.2.7 #define ADC0_SEQA_IRQn ADC_SEQA_IRQn

Definition at line 149 of file cmsis_5411x_m0.h.

#### 6.28.2.8 #define ADC0_SEQB_IRQHandler ADC_SEQB_IRQHandler

Definition at line 161 of file cmsis_5411x_m0.h.

#### 6.28.2.9 #define ADC0_SEQB_IRQn ADC_SEQB_IRQn

Definition at line 150 of file cmsis_5411x_m0.h.

#### 6.28.2.10 #define ADC0_THCMP_IRQHandler ADC_THCMP_IRQHandler

Definition at line 162 of file cmsis_5411x_m0.h.

#### 6.28.2.11 #define ADC0_THCMP_IRQn ADC_THCMP_IRQn

Definition at line 151 of file cmsis_5411x_m0.h.

#### 6.28.2.12 #define SCT_IRQHandler SCT0_IRQHandler

Definition at line 159 of file cmsis_5411x_m0.h.

#### 6.28.2.13 #define SCT_IRQn SCT0_IRQn

Definition at line 148 of file cmsis_5411x_m0.h.

#### 6.28.2.14 #define TIMER0_IRQHandler CT32B0_IRQHandler

Interrupt handler Alias.

Definition at line 154 of file cmsis_5411x_m0.h.

#### 6.28.2.15 #define TIMER0_IRQn CT32B0_IRQn

interrupt Alias

Definition at line 143 of file cmsis_5411x_m0.h.

#### 6.28.2.16 #define TIMER1_IRQHandler CT32B1_IRQHandler

Definition at line 155 of file cmsis_5411x_m0.h.

#### 6.28.2.17 #define TIMER1_IRQn CT32B1_IRQn

Definition at line 144 of file cmsis_5411x_m0.h.

#### 6.28.2.18 #define TIMER2_IRQHandler CT32B2_IRQHandler

Definition at line 156 of file cmsis_5411x_m0.h.

#### 6.28.2.19 #define TIMER2_IRQn CT32B2_IRQn

Definition at line 145 of file cmsis_5411x_m0.h.

#### 6.28.2.20 #define TIMER3_IRQHandler CT32B3_IRQHandler

Definition at line 157 of file cmsis_5411x_m0.h.

#### 6.28.2.21 #define TIMER3_IRQn CT32B3_IRQn

Definition at line 146 of file cmsis_5411x_m0.h.

#### 6.28.2.22 #define TIMER4_IRQHandler CT32B4_IRQHandler

Definition at line 158 of file cmsis_5411x_m0.h.

#### 6.28.2.23 #define TIMER4_IRQn CT32B4_IRQn

Definition at line 147 of file cmsis_5411x_m0.h.

# 6.29 CHIP: LPC5411X M4 core CMSIS include file

## 6.29.1 Detailed Description

### Modules

- CHIP: LPC5411X M4 core Cortex CMSIS definitions
- CHIP_5411X: LPC5411X M4 core peripheral interrupt numbers

# 6.30 CHIP: LPC5411X M4 core Cortex CMSIS definitions

## 6.30.1 Detailed Description

### Macros

- #define __CM4_REV 0x0001
- #define __MPU_PRESENT 1
- #define __NVIC_PRIO_BITS 3
- #define __Vendor_SysTickConfig 0
- #define __FPU_PRESENT 1
- #define TIMER0_IRQn CT32B0_IRQn

  *interrupt Alias*
- #define TIMER1_IRQn CT32B1_IRQn
- #define TIMER2_IRQn CT32B2_IRQn
- #define TIMER3_IRQn CT32B3_IRQn
- #define TIMER4_IRQn CT32B4_IRQn
- #define SCT_IRQn SCT0_IRQn
- #define ADC0_SEQA_IRQn ADC_SEQA_IRQn
- #define ADC0_SEQB_IRQn ADC_SEQB_IRQn
- #define ADC0_THCMP_IRQn ADC_THCMP_IRQn
- #define TIMER0_IRQHandler CT32B0_IRQHandler

  *Interrupt handler Alias.*
- #define TIMER1_IRQHandler CT32B1_IRQHandler
- #define TIMER2_IRQHandler CT32B2_IRQHandler
- #define TIMER3_IRQHandler CT32B3_IRQHandler
- #define TIMER4_IRQHandler CT32B4_IRQHandler
- #define SCT_IRQHandler SCT0_IRQHandler
- #define ADC0_SEQA_IRQHandler ADC_SEQA_IRQHandler
- #define ADC0_SEQB_IRQHandler ADC_SEQB_IRQHandler
- #define ADC0_THCMP_IRQHandler ADC_THCMP_IRQHandler

## 6.30.2 Macro Definition Documentation

### 6.30.2.1 #define __CM4_REV 0x0001

Cortex-M4 Core Revision

Definition at line 146 of file cmsis_5411x.h.

### 6.30.2.2 #define __FPU_PRESENT 1

Definition at line 150 of file cmsis_5411x.h.

### 6.30.2.3 #define __MPU_PRESENT 1

MPU present or not

Definition at line 147 of file cmsis_5411x.h.

### 6.30.2.4 #define __NVIC_PRIO_BITS 3

Number of Bits used for Priority Levels

Definition at line 148 of file cmsis_5411x.h.

#### 6.30.2.5 #define __Vendor_SysTickConfig 0

Set to 1 if different SysTick Config is used

Definition at line 149 of file cmsis_5411x.h.

#### 6.30.2.6 #define ADC0_SEQA_IRQHandler ADC_SEQA_IRQHandler

Definition at line 170 of file cmsis_5411x.h.

#### 6.30.2.7 #define ADC0_SEQA_IRQn ADC_SEQA_IRQn

Definition at line 159 of file cmsis_5411x.h.

#### 6.30.2.8 #define ADC0_SEQB_IRQHandler ADC_SEQB_IRQHandler

Definition at line 171 of file cmsis_5411x.h.

#### 6.30.2.9 #define ADC0_SEQB_IRQn ADC_SEQB_IRQn

Definition at line 160 of file cmsis_5411x.h.

#### 6.30.2.10 #define ADC0_THCMP_IRQHandler ADC_THCMP_IRQHandler

Definition at line 172 of file cmsis_5411x.h.

#### 6.30.2.11 #define ADC0_THCMP_IRQn ADC_THCMP_IRQn

Definition at line 161 of file cmsis_5411x.h.

#### 6.30.2.12 #define SCT_IRQHandler SCT0_IRQHandler

Definition at line 169 of file cmsis_5411x.h.

#### 6.30.2.13 #define SCT_IRQn SCT0_IRQn

Definition at line 158 of file cmsis_5411x.h.

#### 6.30.2.14 #define TIMER0_IRQHandler CT32B0_IRQHandler

Interrupt handler Alias.

Definition at line 164 of file cmsis_5411x.h.

#### 6.30.2.15 #define TIMER0_IRQn CT32B0_IRQn

interrupt Alias

Definition at line 153 of file cmsis_5411x.h.

#### 6.30.2.16 #define TIMER1_IRQHandler CT32B1_IRQHandler

Definition at line 165 of file cmsis_5411x.h.

#### 6.30.2.17 #define TIMER1_IRQn CT32B1_IRQn

Definition at line 154 of file cmsis_5411x.h.

#### 6.30.2.18 #define TIMER2_IRQHandler CT32B2_IRQHandler

Definition at line 166 of file cmsis_5411x.h.

#### 6.30.2.19 #define TIMER2_IRQn CT32B2_IRQn

Definition at line 155 of file cmsis_5411x.h.

#### 6.30.2.20 #define TIMER3_IRQHandler CT32B3_IRQHandler

Definition at line 167 of file cmsis_5411x.h.

#### 6.30.2.21 #define TIMER3_IRQn CT32B3_IRQn

Definition at line 156 of file cmsis_5411x.h.

#### 6.30.2.22 #define TIMER4_IRQHandler CT32B4_IRQHandler

Definition at line 168 of file cmsis_5411x.h.

#### 6.30.2.23 #define TIMER4_IRQn CT32B4_IRQn

Definition at line 157 of file cmsis_5411x.h.

# 6.31 CHIP: LPC5411X Mailbox M4/M0+ driver

## 6.31.1 Detailed Description

### Data Structures

- struct LPC_MBOXIRQ_T
- struct LPC_MBOX_T

### Macros

- #define MAILBOX_AVAIL (MAILBOX_CM4 + 1) /∗ Number of available mailboxes ∗/

### Enumerations

- enum MBOX_IDX_T { MAILBOX_CM0PLUS = 0, MAILBOX_CM4 }

### Functions

- __STATIC_INLINE void Chip_MBOX_Init (LPC_MBOX_T ∗pMBOX)

  *Initialize mailbox.*
- __STATIC_INLINE void Chip_MBOX_DeInit (LPC_MBOX_T ∗pMBOX)

  *Shutdown mailbox.*
- __STATIC_INLINE void Chip_MBOX_SetValue (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id, uint32_t mboxData)

  *Set data value in the mailbox based on the CPU ID.*
- __STATIC_INLINE void Chip_MBOX_SetValueBits (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id, uint32_t mboxSetBits)

  *Set data bits in the mailbox based on the CPU ID.*
- __STATIC_INLINE void Chip_MBOX_ClearValueBits (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id, uint32_t mboxClrBits)

  *Clear data bits in the mailbox based on the CPU ID.*
- __STATIC_INLINE uint32_t Chip_MBOX_GetValue (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id)

  *Get data in the mailbox based on the cpu_id.*
- __STATIC_INLINE uint32_t Chip_MBOX_GetMutex (LPC_MBOX_T ∗pMBOX)

  *Get MUTEX state and lock mutex.*
- __STATIC_INLINE void Chip_MBOX_SetMutex (LPC_MBOX_T ∗pMBOX)

  *Set MUTEX state.*

## 6.31.2 Macro Definition Documentation

### 6.31.2.1 #define MAILBOX_AVAIL (MAILBOX_CM4 + 1) /∗ Number of available mailboxes ∗/

Definition at line 49 of file mailbox_5411x.h.

## 6.31.3 Enumeration Type Documentation

### 6.31.3.1 enum MBOX_IDX_T

**Enumerator**

***MAILBOX_CM0PLUS***

***MAILBOX_CM4***

Definition at line 45 of file mailbox_5411x.h.

## 6.31.4 Function Documentation

### 6.31.4.1 __STATIC_INLINE void Chip_MBOX_ClearValueBits ( LPC_MBOX_T ∗ *pMBOX,* uint32_t *cpu_id,* uint32_t *mboxClrBits* )

Clear data bits in the mailbox based on the CPU ID.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |
| *cpu_id* | : MAILBOX_CM0PLUS is M0+ or MAILBOX_CM4 is M4 |
| *mboxClrBits* | : data bits to clear in the mailbox |

**Returns**

Nothing

**Note**

Clear data bits to send via the MBOX to the other core. A value of 0 will do nothing. Only clears bits selected with a 1 in it's bit position.

Definition at line 124 of file mailbox_5411x.h.

### 6.31.4.2 __STATIC_INLINE void Chip_MBOX_DeInit ( LPC_MBOX_T ∗ *pMBOX* )

Shutdown mailbox.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |

**Returns**

Nothing

Definition at line 83 of file mailbox_5411x.h.

### 6.31.4.3 __STATIC_INLINE uint32_t Chip_MBOX_GetMutex ( LPC_MBOX_T ∗ *pMBOX* )

Get MUTEX state and lock mutex.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |

**Returns**

See note

**Note**

Returns '1' if the mutex was taken or '0' if another resources has the mutex locked. Once a mutex is taken, it can be returned with the Chip_MBOX_SetMutex() function.

Definition at line 148 of file mailbox_5411x.h.

#### 6.31.4.4 __STATIC_INLINE uint32_t Chip_MBOX_GetValue ( LPC_MBOX_T ∗ *pMBOX,* uint32_t *cpu_id* )

Get data in the mailbox based on the cpu_id.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |
| *cpu_id* | : MAILBOX_CM0PLUS is M0+ or MAILBOX_CM4 is M4 |

**Returns**

Current mailbox data

Definition at line 135 of file mailbox_5411x.h.

#### 6.31.4.5 __STATIC_INLINE void Chip_MBOX_Init ( LPC_MBOX_T ∗ *pMBOX* )

Initialize mailbox.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |

**Returns**

Nothing

**Note**

Even if both cores use the amilbox, only 1 core should initialize it.

Definition at line 73 of file mailbox_5411x.h.

#### 6.31.4.6 __STATIC_INLINE void Chip_MBOX_SetMutex ( LPC_MBOX_T ∗ *pMBOX* )

Set MUTEX state.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |

**Returns**

Nothing

**Note**

Sets mutex state to '1' and allows other resources to get the mutex

Definition at line 159 of file mailbox_5411x.h.

#### 6.31.4.7 __STATIC_INLINE void Chip_MBOX_SetValue ( LPC_MBOX_T ∗ *pMBOX,* uint32_t *cpu_id,* uint32_t *mboxData* )

Set data value in the mailbox based on the CPU ID.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |
| *cpu_id* | : MAILBOX_CM0PLUS is M0+ or MAILBOX_CM4 is M4 |
| *mboxData* | : data to send in the mailbox |

**Returns**

Nothing

**Note**

Sets a data value to send via the MBOX to the other core.

Definition at line 96 of file mailbox_5411x.h.

#### 6.31.4.8 __STATIC_INLINE void Chip_MBOX_SetValueBits ( LPC_MBOX_T ∗ *pMBOX,* uint32_t *cpu_id,* uint32_t *mboxSetBits* )

Set data bits in the mailbox based on the CPU ID.

**Parameters**

| | |
|---|---|
| *pMBOX* | : Pointer to the mailbox register structure |
| *cpu_id* | : MAILBOX_CM0PLUS is M0+ or MAILBOX_CM4 is M4 |
| *mboxSetBits* | : data bits to set in the mailbox |

**Returns**

Nothing

**Note**

Sets data bits to send via the MBOX to the other core, A value of 0 will do nothing. Only sets bits selected with a 1 in it's bit position.

Definition at line 110 of file mailbox_5411x.h.

# 6.32 CHIP: LPC5411X Micro Tick driver

## 6.32.1 Detailed Description

### Data Structures

- struct LPC_UTICK_T

  *Micro Tick register block structure.*

### Macros

- #define UTICK_CTRL_REPEAT ((uint32_t) 1UL << 31)

  *UTick register definitions.*
- #define UTICK_CTRL_DELAY_MASK ((uint32_t) 0x7FFFFFFF)
- #define UTICK_STATUS_INTR ((uint32_t) 1 << 0)
- #define UTICK_STATUS_ACTIVE ((uint32_t) 1 << 1)
- #define UTICK_STATUS_MASK ((uint32_t) 0x03)

### Functions

- __STATIC_INLINE void Chip_UTICK_Init (LPC_UTICK_T ∗pUTICK)

  *Initialize the UTICK peripheral.*
- __STATIC_INLINE void Chip_UTICK_DeInit (LPC_UTICK_T ∗pUTICK)

  *De-initialize the UTICK peripheral.*
- __STATIC_INLINE void Chip_UTICK_SetTick (LPC_UTICK_T ∗pUTICK, uint32_t tick_value, bool repeat)

  *Setup UTICK.*
- __STATIC_INLINE void Chip_UTICK_SetDelayMs (LPC_UTICK_T ∗pUTICK, uint32_t delayMs, bool repeat)

  *Setup UTICK for the passed delay (in mS)*
- __STATIC_INLINE uint32_t Chip_UTICK_GetTick (LPC_UTICK_T ∗pUTICK)

  *Read UTICK Value.*
- __STATIC_INLINE void Chip_UTICK_Halt (LPC_UTICK_T ∗pUTICK)

  *Halt UTICK timer.*
- __STATIC_INLINE uint32_t Chip_UTICK_GetStatus (LPC_UTICK_T ∗pUTICK)

  *Returns the status of UTICK.*
- __STATIC_INLINE void Chip_UTICK_ClearInterrupt (LPC_UTICK_T ∗pUTICK)

  *Clears UTICK Interrupt flag.*

## 6.32.2 Macro Definition Documentation

### 6.32.2.1 #define UTICK_CTRL_DELAY_MASK ((uint32_t) 0x7FFFFFFF)

UTick Delay Value Mask

Definition at line 58 of file utick_5411x.h.

### 6.32.2.2 #define UTICK_CTRL_REPEAT ((uint32_t) 1UL << 31)

UTick register definitions.

UTick repeat delay bit

Definition at line 56 of file utick_5411x.h.

#### 6.32.2.3 #define UTICK_STATUS_ACTIVE ((uint32_t) 1 << 1)

UTick Active Status bit

Definition at line 62 of file utick_5411x.h.

#### 6.32.2.4 #define UTICK_STATUS_INTR ((uint32_t) 1 << 0)

UTick Interrupt Status bit

Definition at line 60 of file utick_5411x.h.

#### 6.32.2.5 #define UTICK_STATUS_MASK ((uint32_t) 0x03)

UTick Status Register Mask

Definition at line 64 of file utick_5411x.h.

### 6.32.3 Function Documentation

#### 6.32.3.1 __STATIC_INLINE void Chip_UTICK_ClearInterrupt ( LPC_UTICK_T ∗ *pUTICK* )

Clears UTICK Interrupt flag.

**Parameters**

| | |
|---|---|
| *pUTICK* | : The base address of UTICK block |

**Returns**

Nothing

Definition at line 161 of file utick_5411x.h.

#### 6.32.3.2 __STATIC_INLINE void Chip_UTICK_DeInit ( LPC_UTICK_T ∗ *pUTICK* )

De-initialize the UTICK peripheral.

**Parameters**

| | |
|---|---|
| *pUTICK* | : UTICK peripheral selected |

**Returns**

Nothing

Definition at line 82 of file utick_5411x.h.

#### 6.32.3.3 __STATIC_INLINE uint32_t Chip_UTICK_GetStatus ( LPC_UTICK_T ∗ *pUTICK* )

Returns the status of UTICK.

**Parameters**

| | |
|---|---|
| *pUTICK* | : The base address of UTICK block |

**Returns**

Micro tick timer status register value

Definition at line 151 of file utick_5411x.h.

#### 6.32.3.4 __STATIC_INLINE uint32_t Chip_UTICK_GetTick ( LPC_UTICK_T ∗ *pUTICK* )

Read UTICK Value.

**Parameters**

| | |
|---|---|
| *pUTICK* | : The base address of UTICK block |

**Returns**

Current tick value

Definition at line 131 of file utick_5411x.h.

#### 6.32.3.5 __STATIC_INLINE void Chip_UTICK_Halt ( LPC_UTICK_T ∗ *pUTICK* )

Halt UTICK timer.

**Parameters**

| | |
|---|---|
| *pUTICK* | : The base address of UTICK block |

**Returns**

Nothing

Definition at line 141 of file utick_5411x.h.

#### 6.32.3.6 __STATIC_INLINE void Chip_UTICK_Init ( LPC_UTICK_T ∗ *pUTICK* )

Initialize the UTICK peripheral.

**Parameters**

| | |
|---|---|
| *pUTICK* | : UTICK peripheral selected |

**Returns**

Nothing

Definition at line 71 of file utick_5411x.h.

#### 6.32.3.7 __STATIC_INLINE void Chip_UTICK_SetDelayMs ( LPC_UTICK_T ∗ *pUTICK,* uint32_t *delayMs,* bool *repeat* )

Setup UTICK for the passed delay (in mS)

**Parameters**

| | |
|---|---|
| *pUTICK* | : The base address of UTICK block |
| *delayMs* | : Delay value in mS (Maximum is 1000mS) |
| *repeat* | : If true then delay repeats continuously else it is one time |

**Returns**

Nothing

**Note**

The WDT oscillator runs at about 500KHz, so delays in uS won't be too accurate.

Definition at line 112 of file utick_5411x.h.

### 6.32.3.8 __STATIC_INLINE void Chip_UTICK_SetTick ( LPC_UTICK_T ∗ *pUTICK,* uint32_t *tick_value,* bool *repeat* )

Setup UTICK.

**Parameters**

| | |
|---|---|
| *pUTICK* | : The base address of UTICK block |
| *tick_value* | : Tick value, should not exceed UTICK_CTRL_DELAY_MASK |
| *repeat* | : If true then delay repeats continuously else it is one time |

**Returns**

Nothing

Definition at line 94 of file utick_5411x.h.

# 6.33 CHIP: LPC5411X Multi-Rate Timer driver

## 6.33.1 Detailed Description

### Data Structures

- struct LPC_MRT_CH_T

  *MRT register block structure.*
- struct LPC_MRT_T

  *MRT register block structure.*

### Macros

- #define MRT_CHANNELS_NUM (4)

  *LPC5411X MRT chip configuration.*
- #define MRT_NO_IDLE_CHANNEL (0x40)
- #define MRT_INTVAL_IVALUE (0xFFFFFF) /∗ Maximum interval load value and mask ∗/

  *MRT register bit fields & masks.*
- #define MRT_INTVAL_LOAD (0x80000000UL) /∗ Force immediate load of timer interval register bit ∗/
- #define MRT_CTRL_INTEN_MASK (0x01)
- #define MRT_CTRL_MODE_MASK (0x06)
- #define MRT_STAT_INTFLAG (0x01)
- #define MRT_STAT_RUNNING (0x02)
- #define LPC_MRT_CH0 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[0])
- #define LPC_MRT_CH1 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[1])
- #define LPC_MRT_CH2 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[2])
- #define LPC_MRT_CH3 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[3])
- #define LPC_MRT_CH(ch) ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[(ch)])
- #define MRT0_INTFLAG (1)
- #define MRT1_INTFLAG (2)
- #define MRT2_INTFLAG (4)
- #define MRT3_INTFLAG (8)
- #define MRTn_INTFLAG(ch) (1 << (ch))

### Enumerations

- enum MRT_MODE_T { MRT_MODE_REPEAT = (0 << 1), MRT_MODE_ONESHOT = (1 << 1) }

  *MRT Interrupt Modes enum.*

### Functions

- __STATIC_INLINE void Chip_MRT_Init (void)

  *Initializes the MRT.*
- __STATIC_INLINE void Chip_MRT_DeInit (void)

  *De-initializes the MRT Channel.*
- __STATIC_INLINE LPC_MRT_CH_T ∗ Chip_MRT_GetRegPtr (uint8_t ch)

  *Returns a pointer to the register block for a MRT channel.*
- __STATIC_INLINE uint32_t Chip_MRT_GetInterval (LPC_MRT_CH_T ∗pMRT)

  *Returns the timer time interval value.*
- __STATIC_INLINE void Chip_MRT_SetInterval (LPC_MRT_CH_T ∗pMRT, uint32_t interval)

  *Sets the timer time interval value.*

- __STATIC_INLINE uint32_t Chip_MRT_GetTimer (LPC_MRT_CH_T ∗pMRT)

  *Returns the current timer value.*
- __STATIC_INLINE bool Chip_MRT_GetEnabled (LPC_MRT_CH_T ∗pMRT)

  *Returns true if the timer is enabled.*
- __STATIC_INLINE void Chip_MRT_SetEnabled (LPC_MRT_CH_T ∗pMRT)

  *Enables the timer.*
- __STATIC_INLINE void Chip_MRT_SetDisabled (LPC_MRT_CH_T ∗pMRT)

  *Disables the timer.*
- __STATIC_INLINE MRT_MODE_T Chip_MRT_GetMode (LPC_MRT_CH_T ∗pMRT)

  *Returns the timer mode (repeat or one-shot)*
- __STATIC_INLINE void Chip_MRT_SetMode (LPC_MRT_CH_T ∗pMRT, MRT_MODE_T mode)

  *Sets the timer mode (repeat or one-shot)*
- __STATIC_INLINE bool Chip_MRT_IsRepeatMode (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer is configured in repeat mode.*
- __STATIC_INLINE bool Chip_MRT_IsOneShotMode (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer is configured in one-shot mode.*
- __STATIC_INLINE bool Chip_MRT_IntPending (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer has an interrupt pending.*
- __STATIC_INLINE void Chip_MRT_IntClear (LPC_MRT_CH_T ∗pMRT)

  *Clears the pending interrupt (if any)*
- __STATIC_INLINE bool Chip_MRT_Running (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer is running.*
- __STATIC_INLINE uint8_t Chip_MRT_GetIdleChannel (void)

  *Returns the IDLE channel value.*
- __STATIC_INLINE uint8_t Chip_MRT_GetIdleChannelShifted (void)

  *Returns the IDLE channel value.*
- __STATIC_INLINE uint32_t Chip_MRT_GetIntPending (void)

  *Returns the interrupt pending status for all MRT channels.*
- __STATIC_INLINE bool Chip_MRT_GetIntPendingByChannel (uint8_t ch)

  *Returns the interrupt pending status for a singel MRT channel.*
- __STATIC_INLINE void Chip_MRT_ClearIntPending (uint32_t mask)

  *Clears the interrupt pending status for one or more MRT channels.*

## 6.33.2 Macro Definition Documentation

### 6.33.2.1 #define LPC_MRT_CH( *ch* ) ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[(ch)])

Definition at line 99 of file mrt_5411x.h.

### 6.33.2.2 #define LPC_MRT_CH0 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[0])

Definition at line 95 of file mrt_5411x.h.

### 6.33.2.3 #define LPC_MRT_CH1 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[1])

Definition at line 96 of file mrt_5411x.h.

### 6.33.2.4 #define LPC_MRT_CH2 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[2])

Definition at line 97 of file mrt_5411x.h.

#### 6.33.2.5 #define LPC_MRT_CH3 ((LPC_MRT_CH_T *) &LPC_MRT->CHANNEL[3])

Definition at line 98 of file mrt_5411x.h.

#### 6.33.2.6 #define MRT0_INTFLAG (1)

Definition at line 102 of file mrt_5411x.h.

#### 6.33.2.7 #define MRT1_INTFLAG (2)

Definition at line 103 of file mrt_5411x.h.

#### 6.33.2.8 #define MRT2_INTFLAG (4)

Definition at line 104 of file mrt_5411x.h.

#### 6.33.2.9 #define MRT3_INTFLAG (8)

Definition at line 105 of file mrt_5411x.h.

#### 6.33.2.10 #define MRT_CHANNELS_NUM (4)

LPC5411X MRT chip configuration.

Definition at line 47 of file mrt_5411x.h.

#### 6.33.2.11 #define MRT_CTRL_INTEN_MASK (0x01)

Definition at line 87 of file mrt_5411x.h.

#### 6.33.2.12 #define MRT_CTRL_MODE_MASK (0x06)

Definition at line 88 of file mrt_5411x.h.

#### 6.33.2.13 #define MRT_INTVAL_IVALUE (0xFFFFFF) /* Maximum interval load value and mask */

MRT register bit fields & masks.

Definition at line 83 of file mrt_5411x.h.

#### 6.33.2.14 #define MRT_INTVAL_LOAD (0x80000000UL) /* Force immediate load of timer interval register bit */

Definition at line 84 of file mrt_5411x.h.

#### 6.33.2.15 #define MRT_NO_IDLE_CHANNEL (0x40)

Definition at line 48 of file mrt_5411x.h.

#### 6.33.2.16 #define MRT_STAT_INTFLAG (0x01)

Definition at line 91 of file mrt_5411x.h.

#### 6.33.2.17 #define MRT_STAT_RUNNING (0x02)

Definition at line 92 of file mrt_5411x.h.

#### 6.33.2.18 #define MRTn_INTFLAG( *ch* ) (1 << (ch))

Definition at line 106 of file mrt_5411x.h.

### 6.33.3 Enumeration Type Documentation

#### 6.33.3.1 enum MRT_MODE_T

MRT Interrupt Modes enum.

**Enumerator**

***MRT_MODE_REPEAT*** MRT Repeat interrupt mode

***MRT_MODE_ONESHOT*** MRT One-shot interrupt mode

Definition at line 74 of file mrt_5411x.h.

### 6.33.4 Function Documentation

#### 6.33.4.1 __STATIC_INLINE void Chip_MRT_ClearIntPending ( uint32_t *mask* )

Clears the interrupt pending status for one or more MRT channels.

**Parameters**

| | |
|---|---|
| *mask* | : Channels to clear (bit 0 = MRT0, bit 1 = MRT1, etc.) |

**Returns**

Nothing

**Note**

Use this function to clear multiple interrupt pending states in a single call via the IRQ_FLAG register. Performs the same function for all MRT channels in a single call as the Chip_MRT_IntClear() does for a single channel.

Definition at line 327 of file mrt_5411x.h.

#### 6.33.4.2 __STATIC_INLINE void Chip_MRT_DeInit ( void )

De-initializes the MRT Channel.

**Returns**

Nothing

Definition at line 125 of file mrt_5411x.h.

#### 6.33.4.3 __STATIC_INLINE bool Chip_MRT_GetEnabled ( LPC_MRT_CH_T ∗ *pMRT* )

Returns true if the timer is enabled.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

True if enabled, Flase if not enabled

Definition at line 182 of file mrt_5411x.h.

#### 6.33.4.4 __STATIC_INLINE uint8_t Chip_MRT_GetIdleChannel ( void )

Returns the IDLE channel value.

**Returns**

IDLE channel value (unshifted in bits 7..4)

Definition at line 285 of file mrt_5411x.h.

#### 6.33.4.5 __STATIC_INLINE uint8_t Chip_MRT_GetIdleChannelShifted ( void )

Returns the IDLE channel value.

**Returns**

IDLE channel value (shifted in bits 3..0)

Definition at line 294 of file mrt_5411x.h.

#### 6.33.4.6 __STATIC_INLINE uint32_t Chip_MRT_GetInterval ( LPC_MRT_CH_T * *pMRT* )

Returns the timer time interval value.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

Timer time interval value (IVALUE)

Definition at line 146 of file mrt_5411x.h.

#### 6.33.4.7 __STATIC_INLINE uint32_t Chip_MRT_GetIntPending ( void )

Returns the interrupt pending status for all MRT channels.

**Returns**

IRQ pending channel bitfield(bit 0 = MRT0, bit 1 = MRT1, etc.)

Definition at line 303 of file mrt_5411x.h.

#### 6.33.4.8 __STATIC_INLINE bool Chip_MRT_GetIntPendingByChannel ( uint8_t *ch* )

Returns the interrupt pending status for a singel MRT channel.

**Parameters**

| | |
|---|---|
| *ch* | : Channel to check pending interrupt status for |

**Returns**

IRQ pending channel number

Definition at line 313 of file mrt_5411x.h.

#### 6.33.4.9 __STATIC_INLINE MRT_MODE_T Chip_MRT_GetMode ( LPC_MRT_CH_T ∗ *pMRT* )

Returns the timer mode (repeat or one-shot)

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

The current timer mode

Definition at line 212 of file mrt_5411x.h.

#### 6.33.4.10 __STATIC_INLINE LPC_MRT_CH_T∗ Chip_MRT_GetRegPtr ( uint8_t *ch* )

Returns a pointer to the register block for a MRT channel.

**Parameters**

| | |
|---|---|
| *ch* | : MRT channel tog et register block for (0..3) |

**Returns**

Pointer to the MRT register block for the channel

Definition at line 136 of file mrt_5411x.h.

#### 6.33.4.11 __STATIC_INLINE uint32_t Chip_MRT_GetTimer ( LPC_MRT_CH_T ∗ *pMRT* )

Returns the current timer value.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

The current timer value

Definition at line 172 of file mrt_5411x.h.

#### 6.33.4.12 __STATIC_INLINE void Chip_MRT_Init ( void )

Initializes the MRT.

**Returns**

Nothing

Definition at line 112 of file mrt_5411x.h.

#### 6.33.4.13 __STATIC_INLINE void Chip_MRT_IntClear ( LPC_MRT_CH_T ∗ *pMRT* )

Clears the pending interrupt (if any)

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

Nothing

Definition at line 266 of file mrt_5411x.h.

#### 6.33.4.14 __STATIC_INLINE bool Chip_MRT_IntPending ( LPC_MRT_CH_T ∗ *pMRT* )

Check if the timer has an interrupt pending.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

True if interrupt is pending, False if no interrupt is pending

Definition at line 256 of file mrt_5411x.h.

#### 6.33.4.15 __STATIC_INLINE bool Chip_MRT_IsOneShotMode ( LPC_MRT_CH_T ∗ *pMRT* )

Check if the timer is configured in one-shot mode.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

True if in one-shot mode, False if in repeat mode

Definition at line 246 of file mrt_5411x.h.

#### 6.33.4.16 __STATIC_INLINE bool Chip_MRT_IsRepeatMode ( LPC_MRT_CH_T ∗ *pMRT* )

Check if the timer is configured in repeat mode.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

True if in repeat mode, False if in one-shot mode

Definition at line 236 of file mrt_5411x.h.

#### 6.33.4.17 __STATIC_INLINE bool Chip_MRT_Running ( LPC_MRT_CH_T ∗ *pMRT* )

Check if the timer is running.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

True if running, False if stopped

Definition at line 276 of file mrt_5411x.h.

#### 6.33.4.18 __STATIC_INLINE void Chip_MRT_SetDisabled ( LPC_MRT_CH_T ∗ *pMRT* )

Disables the timer.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

Nothing

Definition at line 202 of file mrt_5411x.h.

#### 6.33.4.19 __STATIC_INLINE void Chip_MRT_SetEnabled ( LPC_MRT_CH_T ∗ *pMRT* )

Enables the timer.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |

**Returns**

Nothing

Definition at line 192 of file mrt_5411x.h.

#### 6.33.4.20 __STATIC_INLINE void Chip_MRT_SetInterval ( LPC_MRT_CH_T ∗ *pMRT,* uint32_t *interval* )

Sets the timer time interval value.

**Parameters**

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |
| *interval* | : The interval timeout (31-bits) |

**Returns**

Nothing

**Note**

Setting bit 31 in timer time interval register causes the time interval value to load immediately, otherwise the time interval value will be loaded in next timer cycle.
Example: Chip_MRT_SetInterval(pMRT, 0x500 | MRT_INTVAL_LOAD); // Will load timer interval immediately
Example: Chip_MRT_SetInterval(pMRT, 0x500); // Will load timer interval after internal expires

Definition at line 162 of file mrt_5411x.h.

#### 6.33.4.21 __STATIC_INLINE void Chip_MRT_SetMode ( LPC_MRT_CH_T ∗ *pMRT,* MRT_MODE_T *mode* )

Sets the timer mode (repeat or one-shot)

#### Parameters

| | |
|---|---|
| *pMRT* | : Pointer to selected MRT Channel |
| *mode* | : Timer mode |

#### Returns

Nothing

Definition at line 223 of file mrt_5411x.h.

# 6.34 CHIP: LPC5411X PLL Driver

## 6.34.1 Detailed Description

The PLL in the LPC5411x is flexible, but can be complex to use. This driver provides functions to help setup and use the PLL in it's various supported modes.

This driver does not alter PLL clock source or system clocks outside the PLL (like the main clock source) that may be referenced from the PLL. It may optionally setup system voltages, wait for PLL lock, and power cycle the PLL during setup based on setup flags.

The driver works by first generating a PLL setup structure from a desired PLL configuration structure. The PLL setup structure is then passed to the PLL setup function to setup the PLL. In a user spplication, the PLL setup structure can be pre-populated with PLL setup data to avoid using the PLL configuration structure (or multiple PLL setup structures can be used to more dynamically control PLL output rate).

**How to use this driver**

```
// Setup PLL configuration
PLL_CONFIG_T pllConfig = {
 75000000,  // desiredRate = 75MHz
 0,    // InputRate = 0Hz (not used)
 0    // No flags, function will determine best setup to get closest rate
};

// Get closest PLL setup to get the desired configuration
PLL_SETUP_T pllSetup;
uint32_t actualPllRate;
PLL_ERROR_T pllError;
pllError = Chip_Clock_SetupPLLData(&pllConfig, &pllSetup, &actualPllRate);
if (pllError != PLL_ERROR_SUCCESS) {
 printf("PLL setup error #%x\r\n", (uint32_t) pllError);
 while (1);
}
else {
 printf("PLL config successful, actual config rate = %uHz\r\n", actualPllRate);
}

// Make sure main system clock is not using PLL, as the PLL setup
// function will power off and optionally power on the PLL
Chip_Clock_SetMainClockSource(SYSCON_MAINCLKSRC_IRC);

// Setup PLL source
Chip_Clock_SetSystemPLLSource(SYSCON_PLLCLKSRC_IRC);

// Now to apply the configuration to the PLL
pllSetup.flags = PLL_SETUPFLAG_WAITLOCK;
Chip_Clock_SetupSystemPLLPrec(&pllSetup);

// Switch main system clock to PLL
Chip_Clock_SetMainClockSource(SYSCON_MAINCLKSRC_PLLOUT);
```

### Data Structures

- struct PLL_CONFIG_T

  *PLL configuration structure This structure can be used to configure the settings for a PLL setup structure. Fill in the desired configuration for the PLL and call the PLL setup function to fill in a PLL setup structure.*

- struct PLL_SETUP_T

  *PLL setup structure This structure can be used to pre-build a PLL setup configuration at run-time and quickly set the PLL to the configuration. It can be populated with the PLL setup function. If powering up or waiting for PLL lock, the PLL input clock source should be configured prior to PLL setup.*

## Macros

- #define PLL_CONFIGFLAG_USEINRATE (1 << 0)

  *PLL configuration structure flags for 'flags' field These flags control how the PLL configuration function sets up the PLL setup structure.*

- #define PLL_CONFIGFLAG_FORCENOFRACT (1 << 2)
- #define PLL_SETUPFLAG_POWERUP (1 << 0)

  *PLL setup structure flags for 'flags' field These flags control how the PLL setup function sets up the PLL.*
- #define PLL_SETUPFLAG_WAITLOCK (1 << 1)
- #define PLL_SETUPFLAG_ADGVOLT (1 << 2)

## Enumerations

- enum CHIP_SYSCON_PLLCLKSRC_T {
  SYSCON_PLLCLKSRC_FRO12MHZ = 0, SYSCON_PLLCLKSRC_CLKIN, SYSCON_PLLCLKSRC_WD↩
  T, SYSCON_PLLCLKSRC_RTC,
  SYSCON_PLLCLKSRC_DISABLED = 7 }
- enum SS_PROGMODFM_T {
  SS_MF_512 = (0 << 20), SS_MF_384 = (1 << 20), SS_MF_256 = (2 << 20), SS_MF_128 = (3 << 20),
  SS_MF_64 = (4 << 20), SS_MF_32 = (5 << 20), SS_MF_24 = (6 << 20), SS_MF_16 = (7 << 20) }

  *PLL Spread Spectrum (SS) Programmable modulation frequency See (MF) field in the SYSPLLSSCTRL1 register in the UM.*
- enum SS_PROGMODDP_T {
  SS_MR_K0 = (0 << 23), SS_MR_K1 = (1 << 23), SS_MR_K1_5 = (2 << 23), SS_MR_K2 = (3 << 23),
  SS_MR_K3 = (4 << 23), SS_MR_K4 = (5 << 23), SS_MR_K6 = (6 << 23), SS_MR_K8 = (7 << 23) }

  *PLL Spread Spectrum (SS) Programmable frequency modulation depth See (MR) field in the SYSPLLSSCTRL1 register in the UM.*
- enum SS_MODWVCTRL_T { SS_MC_NOC = (0 << 26), SS_MC_RECC = (2 << 26), SS_MC_MAXC = (3 << 26) }

  *PLL Spread Spectrum (SS) Modulation waveform control See (MC) field in the SYSPLLSSCTRL1 register in the UM. Compensation for low pass filtering of the PLL to get a triangular modulation at the output of the PLL, giving a flat frequency spectrum.*
- enum PLL_ERROR_T {
  PLL_ERROR_SUCCESS = 0, PLL_ERROR_OUTPUT_TOO_LOW, PLL_ERROR_OUTPUT_TOO_HIGH,
  PLL_ERROR_INPUT_TOO_LOW,
  PLL_ERROR_INPUT_TOO_HIGH, PLL_ERROR_OUTSIDE_INTLIMIT }

  *PLL status definitions.*

## Functions

- __STATIC_INLINE void Chip_Clock_SetSystemPLLSource (CHIP_SYSCON_PLLCLKSRC_T src)

  *Set System PLL clock source.*
- uint32_t Chip_Clock_GetSystemPLLInClockRate (void)

  *Return System PLL input clock rate.*
- uint32_t Chip_Clock_GetSystemPLLOutClockRate (bool recompute)

  *Return System PLL output clock rate.*
- void Chip_Clock_SetBypassPLL (bool bypass)

  *Enables and disables PLL bypass mode.*
- __STATIC_INLINE bool Chip_Clock_IsSystemPLLLocked (void)

  *Check if PLL is locked or not.*
- uint32_t Chip_Clock_GetStoredPLLClockRate (void)

  *Get the rate of pll from the stored value.*

- void Chip_Clock_SetStoredPLLClockRate (uint32_t rate)

  *Store the current PLL rate.*
- uint32_t Chip_Clock_GetSystemPLLOutFromSetup (PLL_SETUP_T ∗pSetup)

  *Return System PLL output clock rate from setup structure.*
- PLL_ERROR_T Chip_Clock_SetupPLLData (PLL_CONFIG_T ∗pControl, PLL_SETUP_T ∗pSetup)

  *Set PLL output based on the passed PLL setup data.*
- PLL_ERROR_T Chip_Clock_SetupSystemPLLPrec (PLL_SETUP_T ∗pSetup)

  *Set PLL output from PLL setup structure (precise frequency)*
- PLL_ERROR_T Chip_Clock_SetPLLFreq (const PLL_SETUP_T ∗pSetup)

  *Set PLL output from PLL setup structure (precise frequency)*
- void Chip_Clock_SetupSystemPLL (uint32_t multiply_by, uint32_t input_freq)

  *Set PLL output based on the multiplier and input frequency.*

## 6.34.2 Macro Definition Documentation

### 6.34.2.1 #define PLL_CONFIGFLAG_FORCENOFRACT (1 $<<$ 2)

Force non-fractional output mode, PLL output will not use the fractional, automatic bandwidth, or SS hardware

Definition at line 180 of file pll_5411x.h.

### 6.34.2.2 #define PLL_CONFIGFLAG_USEINRATE (1 $<<$ 0)

PLL configuration structure flags for 'flags' field These flags control how the PLL configuration function sets up the PLL setup structure.

When the PLL_CONFIGFLAG_USEINRATE flag is selected, the 'InputRate' field in the configuration structure must be assigned with the expected PLL frequency. If the PLL_CONFIGFLAG_USEINRATE is not used, 'InputRate' is ignored in the configuration function and the driver will determine the PLL rate from the currently selected PLL source. This flag might be used to configure the PLL input clock more accurately when using the WDT oscillator or a more dyanmic CLKIN source.

When the PLL_CONFIGFLAG_FORCENOFRACT flag is selected, the PLL hardware for the automatic bandwidth selection, Spread Spectrum (SS) support, and fractional M-divider are not used.
Flag to use InputRate in PLL configuration structure for setup

Definition at line 179 of file pll_5411x.h.

### 6.34.2.3 #define PLL_SETUPFLAG_ADGVOLT (1 $<<$ 2)

Optimize system voltage for the new PLL rate

Definition at line 241 of file pll_5411x.h.

### 6.34.2.4 #define PLL_SETUPFLAG_POWERUP (1 $<<$ 0)

PLL setup structure flags for 'flags' field These flags control how the PLL setup function sets up the PLL.

Setup will power on the PLL after setup

Definition at line 239 of file pll_5411x.h.

#### 6.34.2.5 #define PLL_SETUPFLAG_WAITLOCK (1 $<<$ 1)

Setup will wait for PLL lock, implies the PLL will be pwoered on

Definition at line 240 of file pll_5411x.h.

### 6.34.3 Enumeration Type Documentation

#### 6.34.3.1 enum CHIP_SYSCON_PLLCLKSRC_T

Clock sources for system PLLs

**Enumerator**

***SYSCON_PLLCLKSRC_FRO12MHZ*** 12MHz FRO

***SYSCON_PLLCLKSRC_CLKIN*** External clock input pin

***SYSCON_PLLCLKSRC_WDT*** Watchdog oscillator

***SYSCON_PLLCLKSRC_RTC*** RTC 32KHz oscillator

***SYSCON_PLLCLKSRC_DISABLED*** PLL input clock is disabled

Definition at line 100 of file pll_5411x.h.

#### 6.34.3.2 enum PLL_ERROR_T

PLL status definitions.

**Enumerator**

***PLL_ERROR_SUCCESS*** PLL operation was successful

***PLL_ERROR_OUTPUT_TOO_LOW*** PLL output rate request was too low

***PLL_ERROR_OUTPUT_TOO_HIGH*** PLL output rate request was too high

***PLL_ERROR_INPUT_TOO_LOW*** PLL input rate is too low

***PLL_ERROR_INPUT_TOO_HIGH*** PLL input rate is too high

***PLL_ERROR_OUTSIDE_INTLIMIT*** Requested output rate isn't possible

Definition at line 261 of file pll_5411x.h.

#### 6.34.3.3 enum SS_MODWVCTRL_T

PLL Spread Spectrum (SS) Modulation waveform control See (MC) field in the SYSPLLSSCTRL1 register in the UM.
Compensation for low pass filtering of the PLL to get a triangular modulation at the output of the PLL, giving a flat frequency spectrum.

**Enumerator**

***SS_MC_NOC*** no compensation

***SS_MC_RECC*** recommended setting

***SS_MC_MAXC*** max. compensation

Definition at line 215 of file pll_5411x.h.

#### 6.34.3.4 enum SS_PROGMODDP_T

PLL Spread Spectrum (SS) Programmable frequency modulation depth See (MR) field in the SYSPLLSSCTRL1 register in the UM.

**Enumerator**

***SS_MR_K0*** k = 0 (no spread spectrum)
***SS_MR_K1*** k = 1
***SS_MR_K1_5*** k = 1.5
***SS_MR_K2*** k = 2
***SS_MR_K3*** k = 3
***SS_MR_K4*** k = 4
***SS_MR_K6*** k = 6
***SS_MR_K8*** k = 8

Definition at line 199 of file pll_5411x.h.

#### 6.34.3.5 enum SS_PROGMODFM_T

PLL Spread Spectrum (SS) Programmable modulation frequency See (MF) field in the SYSPLLSSCTRL1 register in the UM.

**Enumerator**

***SS_MF_512*** Nss = 512 (fm 3.9 - 7.8 kHz)
***SS_MF_384*** Nss = 384 (fm 5.2 - 10.4 kHz)
***SS_MF_256*** Nss = 256 (fm 7.8 - 15.6 kHz)
***SS_MF_128*** Nss = 128 (fm 15.6 - 31.3 kHz)
***SS_MF_64*** Nss = 64 (fm 32.3 - 64.5 kHz)
***SS_MF_32*** Nss = 32 (fm 62.5- 125 kHz)
***SS_MF_24*** Nss = 24 (fm 83.3- 166.6 kHz)
***SS_MF_16*** Nss = 16 (fm 125- 250 kHz)

Definition at line 185 of file pll_5411x.h.

### 6.34.4 Function Documentation

#### 6.34.4.1 uint32_t Chip_Clock_GetStoredPLLClockRate ( void )

Get the rate of pll from the stored value.

**Returns**

Current rate of the PLL from the storage

Definition at line 726 of file pll_5411x.c.

#### 6.34.4.2 uint32_t Chip_Clock_GetSystemPLLInClockRate ( void )

Return System PLL input clock rate.

**Returns**

System PLL input clock rate

Definition at line 653 of file pll_5411x.c.

#### 6.34.4.3 uint32_t Chip_Clock_GetSystemPLLOutClockRate ( bool *recompute* )

Return System PLL output clock rate.

**Parameters**

| | |
|---|---|
| *recompute* | : Forces a PLL rate recomputation if true |

**Returns**

System PLL output clock rate

**Note**

The PLL rate is cached in the driver in a variable as the rate computation function can take some time to perform. It is recommended to use 'false' with the 'recompute' parameter.

Definition at line 738 of file pll_5411x.c.

#### 6.34.4.4 uint32_t Chip_Clock_GetSystemPLLOutFromSetup ( PLL_SETUP_T ∗ *pSetup* )

Return System PLL output clock rate from setup structure.

**Parameters**

| | |
|---|---|
| *pSetup* | : Pointer to a PLL setup structure |

**Returns**

System PLL output clock rate the setup structure will generate

Definition at line 683 of file pll_5411x.c.

#### 6.34.4.5 __STATIC_INLINE bool Chip_Clock_IsSystemPLLLocked ( void )

Check if PLL is locked or not.

**Returns**

true if the PLL is locked, false if not locked

Definition at line 146 of file pll_5411x.h.

#### 6.34.4.6 void Chip_Clock_SetBypassPLL ( bool *bypass* )

Enables and disables PLL bypass mode.

bypass : true to bypass PLL (PLL output = PLL input, false to disable bypass

**Returns**

System PLL output clock rate

Definition at line 759 of file pll_5411x.c.

#### 6.34.4.7 PLL_ERROR_T Chip_Clock_SetPLLFreq ( const PLL_SETUP_T ∗ *pSetup* )

Set PLL output from PLL setup structure (precise frequency)

**Parameters**

| | |
|---|---|
| *pSetup* | : Pointer to populated PLL setup structure |

**Returns**

PLL_ERROR_SUCCESS on success, or PLL setup error code

**Note**

This function will power off the PLL, setup the PLL with the new setup data, and then optionally powerup the PLL, wait for PLL lock, and adjust system voltages to the new PLL rate. The function will not alter any source clocks (ie, main systen clock) that may use the PLL, so these should be setup prior to and after exiting the function.

Definition at line 835 of file pll_5411x.c.

#### 6.34.4.8 void Chip_Clock_SetStoredPLLClockRate ( uint32_t *rate* )

Store the current PLL rate.

**Parameters**

| | |
|---|---|
| *rate* | Current rate of the PLL |

**Returns**

Nothing

Definition at line 732 of file pll_5411x.c.

#### 6.34.4.9 __STATIC_INLINE void Chip_Clock_SetSystemPLLSource ( CHIP_SYSCON_PLLCLKSRC_T *src* )

Set System PLL clock source.

**Parameters**

| | |
|---|---|
| *src* | : Clock source for system PLL |

**Returns**

Nothing

**Note**

The PLL should be pwoered down prior to changing the source.

Definition at line 114 of file pll_5411x.h.

#### 6.34.4.10 PLL_ERROR_T Chip_Clock_SetupPLLData ( PLL_CONFIG_T ∗ *pControl,* PLL_SETUP_T ∗ *pSetup* )

Set PLL output based on the passed PLL setup data.

**Parameters**

| | |
|---|---|
| *pControl* | : Pointer to populated PLL control structure to generate setup with |
| *pSetup* | : Pointer to PLL setup structure to be filled |

**Returns**

PLL_ERROR_SUCCESS on success, or PLL setup error code

**Note**

Actual frequency for setup may vary from the desired frequency based on the accuracy of input clocks, rounding, non-fractional PLL mode, etc.

Definition at line 770 of file pll_5411x.c.

#### 6.34.4.11 void Chip_Clock_SetupSystemPLL ( uint32_t *multiply_by,* uint32_t *input_freq* )

Set PLL output based on the multiplier and input frequency.

**Parameters**

| | |
|---|---|
| *multiply_by* | : multiplier |
| *input_freq* | : Clock input frequency of the PLL |

**Returns**

Nothing

**Note**

Unlike the Chip_Clock_SetupSystemPLLPrec() function, this function does not disable or enable PLL power, wait for PLL lock, or adjust system voltages. These must be done in the application. The function will not alter any source clocks (ie, main systen clock) that may use the PLL, so these should be setup prior to and after exiting the function.

Definition at line 866 of file pll_5411x.c.

#### 6.34.4.12 PLL_ERROR_T Chip_Clock_SetupSystemPLLPrec ( PLL_SETUP_T ∗ *pSetup* )

Set PLL output from PLL setup structure (precise frequency)

**Parameters**

| | |
|---|---|
| *pSetup* | : Pointer to populated PLL setup structure |

**Returns**

PLL_ERROR_SUCCESS on success, or PLL setup error code

**Note**

This function will power off the PLL, setup the PLL with the new setup data, and then optionally powerup the PLL, wait for PLL lock, and adjust system voltages to the new PLL rate. The function will not alter any source clocks (ie, main systen clock) that may use the PLL, so these should be setup prior to and after exiting the function.

Definition at line 799 of file pll_5411x.c.

# 6.35 CHIP: LPC5411X Peripheral addresses and register set declarations

## 6.35.1 Detailed Description

### Macros

- #define LPC_FLASHMEM_BASE 0x00000000UL
- #define LPC_SRAMX_BASE 0x04000000UL
- #define LPC_SRAM0_BASE 0x20000000UL
- #define LPC_SRAM1_BASE 0x20010000UL
- #define LPC_SRAM2_BASE 0x20018000UL
- #define LPC_ROM_BASE 0x03000000UL
- #define LPC_SYSCON_BASE 0x40000000UL
- #define LPC_IOCON_BASE 0x40001000UL
- #define LPC_GPIO_GROUPINT0_BASE 0x40002000UL
- #define LPC_GPIO_GROUPINT1_BASE 0x40003000UL
- #define LPC_PIN_INT_BASE 0x40004000UL
- #define LPC_INMUX_BASE 0x40005000UL
- #define LPC_TIMER0_BASE 0x40008000UL
- #define LPC_TIMER1_BASE 0x40009000UL
- #define LPC_WWDT_BASE 0x4000C000UL
- #define LPC_MRT_BASE 0x4000D000UL
- #define LPC_UTICK_BASE 0x4000E000UL
- #define LPC_PMU_BASE 0x40020000UL
- #define LPC_TIMER2_BASE 0x40028000UL
- #define LPC_RTC_BASE 0x4002C000UL
- #define LPC_FMC_BASE 0x40034000UL
- #define LPC_ASYNC_SYSCON_BASE 0x40040000UL
- #define LPC_TIMER3_BASE 0x40048000UL
- #define LPC_TIMER4_BASE 0x40049000UL
- #define LPC_SPIFI_BASE 0x40080000UL
- #define LPC_DMA_BASE 0x40082000UL
- #define LPC_USB_BASE 0x40084000UL
- #define LPC_SCT_BASE 0x40085000UL
- #define LPC_FLEXCOMM0_BASE 0x40086000UL
- #define LPC_FLEXCOMM1_BASE 0x40087000UL
- #define LPC_FLEXCOMM2_BASE 0x40088000UL
- #define LPC_FLEXCOMM3_BASE 0x40089000UL
- #define LPC_FLEXCOMM4_BASE 0x4008A000UL
- #define LPC_MBOX_BASE 0x4008B000UL
- #define LPC_GPIO_PORT_BASE 0x4008C000UL
- #define LPC_DMIC_BASE 0x40090000UL
- #define LPC_CRC_BASE 0x40095000UL
- #define LPC_FLEXCOMM5_BASE 0x40096000UL
- #define LPC_FLEXCOMM6_BASE 0x40097000UL
- #define LPC_FLEXCOMM7_BASE 0x40098000UL
- #define LPC_ISPAP_BASE 0x4009C000UL
- #define LPC_ADC_BASE 0x400A0000UL
- #define LPC_GPIO ((LPC_GPIO_T ∗) LPC_GPIO_PORT_BASE)
- #define LPC_DMA ((LPC_DMA_T ∗) LPC_DMA_BASE)
- #define LPC_CRC ((LPC_CRC_T ∗) LPC_CRC_BASE)
- #define LPC_SCT ((LPC_SCT_T ∗) LPC_SCT_BASE)
- #define LPC_MBOX ((LPC_MBOX_T ∗) LPC_MBOX_BASE)
- #define LPC_ADC ((LPC_ADC_T ∗) LPC_ADC_BASE)

- #define LPC_PMU ((LPC_PMU_T ∗) LPC_PMU_BASE)
- #define LPC_DMIC ((LPC_DMIC_T ∗) LPC_DMIC_BASE)
- #define LPC_USB ((LPC_USB_T ∗) LPC_USB_BASE)
- #define LPC_SYSCON ((LPC_SYSCON_T ∗) LPC_SYSCON_BASE)
- #define LPC_TIMER2 ((LPC_TIMER_T ∗) LPC_TIMER2_BASE)
- #define LPC_TIMER3 ((LPC_TIMER_T ∗) LPC_TIMER3_BASE)
- #define LPC_TIMER4 ((LPC_TIMER_T ∗) LPC_TIMER4_BASE)
- #define LPC_GINT ((LPC_GPIOGROUPINT_T ∗) LPC_GPIO_GROUPINT0_BASE)
- #define LPC_PININT ((LPC_PIN_INT_T ∗) LPC_PIN_INT_BASE)
- #define LPC_IOCON ((LPC_IOCON_T ∗) LPC_IOCON_BASE)
- #define LPC_UTICK ((LPC_UTICK_T ∗) LPC_UTICK_BASE)
- #define LPC_WWDT ((LPC_WWDT_T ∗) LPC_WWDT_BASE)
- #define LPC_RTC ((LPC_RTC_T ∗) LPC_RTC_BASE)
- #define LPC_ASYNC_SYSCON ((LPC_ASYNC_SYSCON_T ∗) LPC_ASYNC_SYSCON_BASE)
- #define LPC_TIMER0 ((LPC_TIMER_T ∗) LPC_TIMER0_BASE)
- #define LPC_TIMER1 ((LPC_TIMER_T ∗) LPC_TIMER1_BASE)
- #define LPC_INMUX ((LPC_INMUX_T ∗) LPC_INMUX_BASE)
- #define LPC_MRT ((LPC_MRT_T ∗) LPC_MRT_BASE)

## 6.35.2 Macro Definition Documentation

### 6.35.2.1 #define LPC_ADC ((LPC_ADC_T ∗) LPC_ADC_BASE)

Definition at line 130 of file chip.h.

### 6.35.2.2 #define LPC_ADC_BASE 0x400A0000UL

Definition at line 120 of file chip.h.

### 6.35.2.3 #define LPC_ASYNC_SYSCON ((LPC_ASYNC_SYSCON_T ∗) LPC_ASYNC_SYSCON_BASE)

Definition at line 148 of file chip.h.

### 6.35.2.4 #define LPC_ASYNC_SYSCON_BASE 0x40040000UL

Definition at line 98 of file chip.h.

### 6.35.2.5 #define LPC_CRC ((LPC_CRC_T ∗) LPC_CRC_BASE)

Definition at line 127 of file chip.h.

### 6.35.2.6 #define LPC_CRC_BASE 0x40095000UL

Definition at line 115 of file chip.h.

### 6.35.2.7 #define LPC_DMA ((LPC_DMA_T ∗) LPC_DMA_BASE)

Definition at line 126 of file chip.h.

#### 6.35.2.8 #define LPC_DMA_BASE 0x40082000UL

Definition at line 104 of file chip.h.

#### 6.35.2.9 #define LPC_DMIC ((LPC_DMIC_T ∗) LPC_DMIC_BASE)

Definition at line 132 of file chip.h.

#### 6.35.2.10 #define LPC_DMIC_BASE 0x40090000UL

Definition at line 114 of file chip.h.

#### 6.35.2.11 #define LPC_FLASHMEM_BASE 0x00000000UL

Definition at line 71 of file chip.h.

#### 6.35.2.12 #define LPC_FLEXCOMM0_BASE 0x40086000UL

Definition at line 107 of file chip.h.

#### 6.35.2.13 #define LPC_FLEXCOMM1_BASE 0x40087000UL

Definition at line 108 of file chip.h.

#### 6.35.2.14 #define LPC_FLEXCOMM2_BASE 0x40088000UL

Definition at line 109 of file chip.h.

#### 6.35.2.15 #define LPC_FLEXCOMM3_BASE 0x40089000UL

Definition at line 110 of file chip.h.

#### 6.35.2.16 #define LPC_FLEXCOMM4_BASE 0x4008A000UL

Definition at line 111 of file chip.h.

#### 6.35.2.17 #define LPC_FLEXCOMM5_BASE 0x40096000UL

Definition at line 116 of file chip.h.

#### 6.35.2.18 #define LPC_FLEXCOMM6_BASE 0x40097000UL

Definition at line 117 of file chip.h.

#### 6.35.2.19 #define LPC_FLEXCOMM7_BASE 0x40098000UL

Definition at line 118 of file chip.h.

#### 6.35.2.20 #define LPC_FMC_BASE 0x40034000UL

Definition at line 95 of file chip.h.

#### 6.35.2.21 #define LPC_GINT ((LPC_GPIOGROUPINT_T ∗) LPC_GPIO_GROUPINT0_BASE)

Definition at line 140 of file chip.h.

#### 6.35.2.22 #define LPC_GPIO ((LPC_GPIO_T ∗) LPC_GPIO_PORT_BASE)

Definition at line 125 of file chip.h.

#### 6.35.2.23 #define LPC_GPIO_GROUPINT0_BASE 0x40002000UL

Definition at line 81 of file chip.h.

#### 6.35.2.24 #define LPC_GPIO_GROUPINT1_BASE 0x40003000UL

Definition at line 82 of file chip.h.

#### 6.35.2.25 #define LPC_GPIO_PORT_BASE 0x4008C000UL

Definition at line 113 of file chip.h.

#### 6.35.2.26 #define LPC_INMUX ((LPC_INMUX_T ∗) LPC_INMUX_BASE)

Definition at line 151 of file chip.h.

#### 6.35.2.27 #define LPC_INMUX_BASE 0x40005000UL

Definition at line 84 of file chip.h.

#### 6.35.2.28 #define LPC_IOCON ((LPC_IOCON_T ∗) LPC_IOCON_BASE)

Definition at line 142 of file chip.h.

#### 6.35.2.29 #define LPC_IOCON_BASE 0x40001000UL

Definition at line 80 of file chip.h.

#### 6.35.2.30 #define LPC_ISPAP_BASE 0x4009C000UL

Definition at line 119 of file chip.h.

#### 6.35.2.31 #define LPC_MBOX ((LPC_MBOX_T ∗) LPC_MBOX_BASE)

Definition at line 129 of file chip.h.

#### 6.35.2.32 #define LPC_MBOX_BASE 0x4008B000UL

Definition at line 112 of file chip.h.

#### 6.35.2.33 #define LPC_MRT ((LPC_MRT_T ∗) LPC_MRT_BASE)

Definition at line 152 of file chip.h.

#### 6.35.2.34 #define LPC_MRT_BASE 0x4000D000UL

Definition at line 88 of file chip.h.

#### 6.35.2.35 #define LPC_PIN_INT_BASE 0x40004000UL

Definition at line 83 of file chip.h.

#### 6.35.2.36 #define LPC_PININT ((LPC_PIN_INT_T ∗) LPC_PIN_INT_BASE)

Definition at line 141 of file chip.h.

#### 6.35.2.37 #define LPC_PMU ((LPC_PMU_T ∗) LPC_PMU_BASE)

Definition at line 131 of file chip.h.

#### 6.35.2.38 #define LPC_PMU_BASE 0x40020000UL

Definition at line 92 of file chip.h.

#### 6.35.2.39 #define LPC_ROM_BASE 0x03000000UL

Definition at line 76 of file chip.h.

#### 6.35.2.40 #define LPC_RTC ((LPC_RTC_T ∗) LPC_RTC_BASE)

Definition at line 145 of file chip.h.

#### 6.35.2.41 #define LPC_RTC_BASE 0x4002C000UL

Definition at line 94 of file chip.h.

#### 6.35.2.42 #define LPC_SCT ((LPC_SCT_T ∗) LPC_SCT_BASE)

Definition at line 128 of file chip.h.

#### 6.35.2.43 #define LPC_SCT_BASE 0x40085000UL

Definition at line 106 of file chip.h.

#### 6.35.2.44 #define LPC_SPIFI_BASE 0x40080000UL

Definition at line 103 of file chip.h.

#### 6.35.2.45 #define LPC_SRAM0_BASE 0x20000000UL

Definition at line 73 of file chip.h.

#### 6.35.2.46 #define LPC_SRAM1_BASE 0x20010000UL

Definition at line 74 of file chip.h.

#### 6.35.2.47 #define LPC_SRAM2_BASE 0x20018000UL

Definition at line 75 of file chip.h.

#### 6.35.2.48 #define LPC_SRAMX_BASE 0x04000000UL

Definition at line 72 of file chip.h.

#### 6.35.2.49 #define LPC_SYSCON ((LPC_SYSCON_T ∗) LPC_SYSCON_BASE)

Definition at line 136 of file chip.h.

#### 6.35.2.50 #define LPC_SYSCON_BASE 0x40000000UL

Definition at line 79 of file chip.h.

#### 6.35.2.51 #define LPC_TIMER0 ((LPC_TIMER_T ∗) LPC_TIMER0_BASE)

Definition at line 149 of file chip.h.

#### 6.35.2.52 #define LPC_TIMER0_BASE 0x40008000UL

Definition at line 85 of file chip.h.

#### 6.35.2.53 #define LPC_TIMER1 ((LPC_TIMER_T ∗) LPC_TIMER1_BASE)

Definition at line 150 of file chip.h.

#### 6.35.2.54 #define LPC_TIMER1_BASE 0x40009000UL

Definition at line 86 of file chip.h.

#### 6.35.2.55 #define LPC_TIMER2 ((LPC_TIMER_T ∗) LPC_TIMER2_BASE)

Definition at line 137 of file chip.h.

#### 6.35.2.56 #define LPC_TIMER2_BASE 0x40028000UL

Definition at line 93 of file chip.h.

#### 6.35.2.57 #define LPC_TIMER3 ((LPC_TIMER_T ∗) LPC_TIMER3_BASE)

Definition at line 138 of file chip.h.

#### 6.35.2.58 #define LPC_TIMER3_BASE 0x40048000UL

Definition at line 99 of file chip.h.

#### 6.35.2.59 #define LPC_TIMER4 ((LPC_TIMER_T ∗) LPC_TIMER4_BASE)

Definition at line 139 of file chip.h.

#### 6.35.2.60 #define LPC_TIMER4_BASE 0x40049000UL

Definition at line 100 of file chip.h.

#### 6.35.2.61 #define LPC_USB ((LPC_USB_T ∗) LPC_USB_BASE)

Definition at line 133 of file chip.h.

#### 6.35.2.62 #define LPC_USB_BASE 0x40084000UL

Definition at line 105 of file chip.h.

#### 6.35.2.63 #define LPC_UTICK ((LPC_UTICK_T ∗) LPC_UTICK_BASE)

Definition at line 143 of file chip.h.

#### 6.35.2.64 #define LPC_UTICK_BASE 0x4000E000UL

Definition at line 89 of file chip.h.

#### 6.35.2.65 #define LPC_WWDT ((LPC_WWDT_T ∗) LPC_WWDT_BASE)

Definition at line 144 of file chip.h.

#### 6.35.2.66 #define LPC_WWDT_BASE 0x4000C000UL

Definition at line 87 of file chip.h.

## 6.36 CHIP: LPC5411X Pin Interrupt and Pattern Match driver

### 6.36.1 Detailed Description

#### Data Structures

- struct LPC_PIN_INT_T

  *LPC5411X Pin Interrupt and Pattern Match register block structure.*

#### Macros

- #define PININT_ISEL_PMODE_MASK ((uint32_t) 0x00FF)
- #define PININT_PMCTRL_MASK ((uint32_t) 0xFF000003)
- #define PININT_PMCTRL_PMATCH_SEL (1 << 0)
- #define PININT_PMCTRL_RXEV_ENA (1 << 1)
- #define PININT_SRC_BITSOURCE_START 8
- #define PININT_SRC_BITSOURCE_MASK 7
- #define PININT_SRC_BITCFG_START 8
- #define PININT_SRC_BITCFG_MASK 7
- #define PININTCH0 (1 << 0)
- #define PININTCH1 (1 << 1)
- #define PININTCH2 (1 << 2)
- #define PININTCH3 (1 << 3)
- #define PININTCH4 (1 << 4)
- #define PININTCH5 (1 << 5)
- #define PININTCH6 (1 << 6)
- #define PININTCH7 (1 << 7)
- #define PININTCH(ch) (1 << (ch))

#### Enumerations

- enum Chip_PININT_SELECT_T {
  PININTSELECT0 = 0, PININTSELECT1 = 1, PININTSELECT2 = 2, PININTSELECT3 = 3,
  PININTSELECT4 = 4, PININTSELECT5 = 5, PININTSELECT6 = 6, PININTSELECT7 = 7 }
- enum Chip_PININT_BITSLICE_T {
  PININTBITSLICE0 = 0, PININTBITSLICE1 = 1, PININTBITSLICE2 = 2, PININTBITSLICE3 = 3,
  PININTBITSLICE4 = 4, PININTBITSLICE5 = 5, PININTBITSLICE6 = 6, PININTBITSLICE7 = 7 }
- enum Chip_PININT_BITSLICE_CFG_T {
  PININT_PATTERNCONST1 = 0x0, PININT_PATTERNRISING = 0x1, PININT_PATTERNFALLING = 0x2,
  PININT_PATTERNRISINGORFALLING = 0x3,
  PININT_PATTERNHIGH = 0x4, PININT_PATTERNLOW = 0x5, PININT_PATTERNCONST0 = 0x6, PININT_PATTERNEVENT = 0x7 }

#### Functions

- __STATIC_INLINE void Chip_PININT_Init (LPC_PIN_INT_T ∗pPININT)

  *Initialize Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_DeInit (LPC_PIN_INT_T ∗pPININT)

  *De-Initialize Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPinModeEdge (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Configure the pins as edge sensitive in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPinModeLevel (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

*Configure the pins as level sensitive in Pin interrupt block.*

- __STATIC_INLINE uint32_t Chip_PININT_GetPinMode (LPC_PIN_INT_T ∗pPININT)

  *Return current PININT edge or level sensitive interrupt selection state.*
- __STATIC_INLINE uint32_t Chip_PININT_GetHighEnabled (LPC_PIN_INT_T ∗pPININT)

  *Return current PININT rising edge or level interrupt enable state.*
- __STATIC_INLINE void Chip_PININT_EnableIntHigh (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Enable rising edge/level PININT interrupts for pins.*
- __STATIC_INLINE void Chip_PININT_DisableIntHigh (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Disable rising edge/level PININT interrupts for pins.*
- __STATIC_INLINE uint32_t Chip_PININT_GetLowEnabled (LPC_PIN_INT_T ∗pPININT)

  *Return current PININT falling edge or level interrupt active level enable state.*
- __STATIC_INLINE void Chip_PININT_EnableIntLow (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Enable falling edge/level active level PININT interrupts for pins.*
- __STATIC_INLINE void Chip_PININT_DisableIntLow (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Disable low edge/level active level PININT interrupts for pins.*
- __STATIC_INLINE uint32_t Chip_PININT_GetRiseStates (LPC_PIN_INT_T ∗pPININT)

  *Return pin states that have a detected latched rising edge (RISE) state.*
- __STATIC_INLINE void Chip_PININT_ClearRiseStates (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Clears pin states that had a latched rising edge (RISE) state.*
- __STATIC_INLINE uint32_t Chip_PININT_GetFallStates (LPC_PIN_INT_T ∗pPININT)

  *Return pin states that have a detected latched falling edge (FALL) state.*
- __STATIC_INLINE void Chip_PININT_ClearFallStates (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Clears pin states that had a latched falling edge (FALL) state.*
- __STATIC_INLINE uint32_t Chip_PININT_GetIntStatus (LPC_PIN_INT_T ∗pPININT)

  *Get interrupt status from Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_ClearIntStatus (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Clear interrupt status in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPatternMatchSrc (LPC_PIN_INT_T ∗pPININT, Chip_PININT_SELECT_T channelNum, Chip_PININT_BITSLICE_T sliceNum)

  *Set source for pattern match in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPatternMatchConfig (LPC_PIN_INT_T ∗pPININT, Chip_PININT_BITSLICE_T sliceNum, Chip_PININT_BITSLICE_CFG_T slice_cfg, bool end_point)

  *Configure the pattern matcch in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_EnablePatternMatch (LPC_PIN_INT_T ∗pPININT)

  *Enable pattern match interrupts in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_DisablePatternMatch (LPC_PIN_INT_T ∗pPININT)

  *Disable pattern match interrupts in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_EnablePatternMatchRxEv (LPC_PIN_INT_T ∗pPININT)

  *Enable RXEV output in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_DisablePatternMatchRxEv (LPC_PIN_INT_T ∗pPININT)

  *Disable RXEV output in Pin interrupt block.*
- __STATIC_INLINE uint32_t Chip_PININT_GetPatternMatchState (LPC_PIN_INT_T ∗pPININT)

  *Return pattern match state.*

## 6.36.2 Macro Definition Documentation

### 6.36.2.1 #define PININT_ISEL_PMODE_MASK ((uint32_t) 0x00FF)

LPC5411X Pin Interrupt and Pattern match engine register bit fields and macros

Definition at line 69 of file pinint_5411x.h.

#### 6.36.2.2 #define PININT_PMCTRL_MASK ((uint32_t) 0xFF000003)

Definition at line 72 of file pinint_5411x.h.

#### 6.36.2.3 #define PININT_PMCTRL_PMATCH_SEL (1 << 0)

Definition at line 75 of file pinint_5411x.h.

#### 6.36.2.4 #define PININT_PMCTRL_RXEV_ENA (1 << 1)

Definition at line 76 of file pinint_5411x.h.

#### 6.36.2.5 #define PININT_SRC_BITCFG_MASK 7

Definition at line 84 of file pinint_5411x.h.

#### 6.36.2.6 #define PININT_SRC_BITCFG_START 8

Definition at line 83 of file pinint_5411x.h.

#### 6.36.2.7 #define PININT_SRC_BITSOURCE_MASK 7

Definition at line 80 of file pinint_5411x.h.

#### 6.36.2.8 #define PININT_SRC_BITSOURCE_START 8

Definition at line 79 of file pinint_5411x.h.

#### 6.36.2.9 #define PININTCH( *ch* ) (1 << (ch))

Definition at line 97 of file pinint_5411x.h.

#### 6.36.2.10 #define PININTCH0 (1 << 0)

LPC5411X Pin Interrupt channel values

Definition at line 89 of file pinint_5411x.h.

#### 6.36.2.11 #define PININTCH1 (1 << 1)

Definition at line 90 of file pinint_5411x.h.

#### 6.36.2.12 #define PININTCH2 (1 << 2)

Definition at line 91 of file pinint_5411x.h.

#### 6.36.2.13 #define PININTCH3 (1 << 3)

Definition at line 92 of file pinint_5411x.h.

#### 6.36.2.14 #define PININTCH4 (1 $<<$ 4)

Definition at line 93 of file pinint_5411x.h.

#### 6.36.2.15 #define PININTCH5 (1 $<<$ 5)

Definition at line 94 of file pinint_5411x.h.

#### 6.36.2.16 #define PININTCH6 (1 $<<$ 6)

Definition at line 95 of file pinint_5411x.h.

#### 6.36.2.17 #define PININTCH7 (1 $<<$ 7)

Definition at line 96 of file pinint_5411x.h.

### 6.36.3 Enumeration Type Documentation

#### 6.36.3.1 enum Chip_PININT_BITSLICE_CFG_T

LPC5411X Pin Matching Interrupt bit slice configuration enum values

**Enumerator**

***PININT_PATTERNCONST1*** Contributes to product term match

***PININT_PATTERNRISING*** Rising edge

***PININT_PATTERNFALLING*** Falling edge

***PININT_PATTERNRISINGORFALLING*** Rising or Falling edge

***PININT_PATTERNHIGH*** High level

***PININT_PATTERNLOW*** Low level

***PININT_PATTERNCONST0*** Never contributes for match

***PININT_PATTERNEVENT*** Match occurs on event

Definition at line 130 of file pinint_5411x.h.

#### 6.36.3.2 enum Chip_PININT_BITSLICE_T

LPC5411X Pin Matching Interrupt bit slice enum values

**Enumerator**

***PININTBITSLICE0*** PININT Bit slice 0

***PININTBITSLICE1*** PININT Bit slice 1

***PININTBITSLICE2*** PININT Bit slice 2

***PININTBITSLICE3*** PININT Bit slice 3

***PININTBITSLICE4*** PININT Bit slice 4

***PININTBITSLICE5*** PININT Bit slice 5

***PININTBITSLICE6*** PININT Bit slice 6

***PININTBITSLICE7*** PININT Bit slice 7

Definition at line 116 of file pinint_5411x.h.

#### 6.36.3.3 enum Chip_PININT_SELECT_T

LPC5411X Pin Interrupt select enum values

**Enumerator**

- ***PININTSELECT0***
- ***PININTSELECT1***
- ***PININTSELECT2***
- ***PININTSELECT3***
- ***PININTSELECT4***
- ***PININTSELECT5***
- ***PININTSELECT6***
- ***PININTSELECT7***

Definition at line 102 of file pinint_5411x.h.

### 6.36.4 Function Documentation

#### 6.36.4.1 __STATIC_INLINE void Chip_PININT_ClearFallStates ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Clears pin states that had a latched falling edge (FALL) state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins with latched states to clear |

**Returns**

Nothing

Definition at line 305 of file pinint_5411x.h.

#### 6.36.4.2 __STATIC_INLINE void Chip_PININT_ClearIntStatus ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Clear interrupt status in Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pin interrupts to clear (ORed value of PININTCH∗) |

**Returns**

Nothing

Definition at line 326 of file pinint_5411x.h.

#### 6.36.4.3 __STATIC_INLINE void Chip_PININT_ClearRiseStates ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Clears pin states that had a latched rising edge (RISE) state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins with latched states to clear |

**Returns**

Nothing

Definition at line 284 of file pinint_5411x.h.

#### 6.36.4.4 __STATIC_INLINE void Chip_PININT_DeInit ( LPC_PIN_INT_T ∗ *pPININT* )

De-Initialize Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

Nothing

Definition at line 158 of file pinint_5411x.h.

#### 6.36.4.5 __STATIC_INLINE void Chip_PININT_DisableIntHigh ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Disable rising edge/level PININT interrupts for pins.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins to disable (ORed value of PININTCH∗) |

**Returns**

Nothing

Definition at line 228 of file pinint_5411x.h.

#### 6.36.4.6 __STATIC_INLINE void Chip_PININT_DisableIntLow ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Disable low edge/level active level PININT interrupts for pins.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins to disable (ORed value of PININTCH∗) |

**Returns**

Nothing

Definition at line 263 of file pinint_5411x.h.

#### 6.36.4.7 __STATIC_INLINE void Chip_PININT_DisablePatternMatch ( LPC_PIN_INT_T ∗ *pPININT* )

Disable pattern match interrupts in Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

Nothing

Definition at line 395 of file pinint_5411x.h.

#### 6.36.4.8 __STATIC_INLINE void Chip_PININT_DisablePatternMatchRxEv ( LPC_PIN_INT_T ∗ *pPININT* )

Disable RXEV output in Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

Nothing

Definition at line 415 of file pinint_5411x.h.

#### 6.36.4.9 __STATIC_INLINE void Chip_PININT_EnableIntHigh ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Enable rising edge/level PININT interrupts for pins.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins to enable (ORed value of PININTCH∗) |

**Returns**

Nothing

Definition at line 217 of file pinint_5411x.h.

#### 6.36.4.10 __STATIC_INLINE void Chip_PININT_EnableIntLow ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Enable falling edge/level active level PININT interrupts for pins.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins to enable (ORed value of PININTCH∗) |

**Returns**

Nothing

Definition at line 252 of file pinint_5411x.h.

#### 6.36.4.11 __STATIC_INLINE void Chip_PININT_EnablePatternMatch ( LPC_PIN_INT_T ∗ *pPININT* )

Enable pattern match interrupts in Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

Nothing

Definition at line 385 of file pinint_5411x.h.

#### 6.36.4.12 __STATIC_INLINE void Chip_PININT_EnablePatternMatchRxEv ( LPC_PIN_INT_T ∗ *pPININT* )

Enable RXEV output in Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

Nothing

Definition at line 405 of file pinint_5411x.h.

#### 6.36.4.13 __STATIC_INLINE uint32_t Chip_PININT_GetFallStates ( LPC_PIN_INT_T ∗ *pPININT* )

Return pin states that have a detected latched falling edge (FALL) state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

PININT states (bit n = high) with a latched rise state detected

Definition at line 294 of file pinint_5411x.h.

#### 6.36.4.14 __STATIC_INLINE uint32_t Chip_PININT_GetHighEnabled ( LPC_PIN_INT_T ∗ *pPININT* )

Return current PININT rising edge or level interrupt enable state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

A bifield containing the rising edge/level enable for each interrupt. Bit 0 = PININT0, 1 = PININT1, etc. For each bit, a 0 means the rising edge/level interrupt is disabled, while a 1 means it's enabled.

Definition at line 206 of file pinint_5411x.h.

#### 6.36.4.15 __STATIC_INLINE uint32_t Chip_PININT_GetIntStatus ( LPC_PIN_INT_T ∗ *pPININT* )

Get interrupt status from Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

> Interrupt status (bit n for PININTn = high means interrupt ie pending)

Definition at line 315 of file pinint_5411x.h.

### 6.36.4.16 __STATIC_INLINE uint32_t Chip_PININT_GetLowEnabled ( LPC_PIN_INT_T ∗ *pPININT* )

Return current PININT falling edge or level interrupt active level enable state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

> A bifield containing the falling edge/level interrupt active level enable for each interrupt. Bit 0 = PININT0, 1 = PININT1, etc. For each bit, a 0 means the falling edge is disabled/level interrupt active low is enabled, while a 1 means the falling edge is enabled/level interrupt active high is enabled.

Definition at line 241 of file pinint_5411x.h.

### 6.36.4.17 __STATIC_INLINE uint32_t Chip_PININT_GetPatternMatchState ( LPC_PIN_INT_T ∗ *pPININT* )

Return pattern match state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

> 8 bit pattern match state, where a 1 in any bit indicates that the corresponding product term has matched by the current state of its inputs.

Definition at line 427 of file pinint_5411x.h.

### 6.36.4.18 __STATIC_INLINE uint32_t Chip_PININT_GetPinMode ( LPC_PIN_INT_T ∗ *pPININT* )

Return current PININT edge or level sensitive interrupt selection state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

> A bifield containing the edge/level sensitive selection for each interrupt. Bit 0 = PININT0, 1 = PININT1, etc. For each bit, a 0 means the edge sensitive interrupt is selected, while a 1 means the level sensitive interrupt is selected.

Definition at line 193 of file pinint_5411x.h.

### 6.36.4.19 __STATIC_INLINE uint32_t Chip_PININT_GetRiseStates ( LPC_PIN_INT_T ∗ *pPININT* )

Return pin states that have a detected latched rising edge (RISE) state.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

PININT states (bit n = high) with a latched rise state detected

Definition at line 273 of file pinint_5411x.h.

### 6.36.4.20 __STATIC_INLINE void Chip_PININT_Init ( LPC_PIN_INT_T ∗ *pPININT* )

Initialize Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |

**Returns**

Nothing

**Note**

This function should be used after the Chip_GPIO_Init() function.

Definition at line 147 of file pinint_5411x.h.

### 6.36.4.21 __STATIC_INLINE void Chip_PININT_SetPatternMatchConfig ( LPC_PIN_INT_T ∗ *pPININT*, Chip_PININT_BITSLICE_T *sliceNum*, Chip_PININT_BITSLICE_CFG_T *slice_cfg*, bool *end_point* )

Configure the pattern matcch in Pin interrupt block.

**Parameters**

| | |
|---|---|
| *pPININT* | : The base address of Pin interrupt block |
| *sliceNum* | : PININT slice number |
| *slice_cfg* | : PININT slice configuration value (enum Chip_PININT_BITSLICE_CFG_T) |
| *end_point* | : If true, current slice is final component |

**Returns**

Nothing

Definition at line 357 of file pinint_5411x.h.

### 6.36.4.22 __STATIC_INLINE void Chip_PININT_SetPatternMatchSrc ( LPC_PIN_INT_T ∗ *pPININT*, Chip_PININT_SELECT_T *channelNum*, Chip_PININT_BITSLICE_T *sliceNum* )

Set source for pattern match in Pin interrupt block.

**Parameters**

| | |
|---:|---|
| *pPININT* | : The base address of Pin interrupt block |
| *channelNum* | : PININT channel number (From 0 to 7) |
| *sliceNum* | : PININT slice number |

**Returns**

Nothing

Definition at line 338 of file pinint_5411x.h.

#### 6.36.4.23 __STATIC_INLINE void Chip_PININT_SetPinModeEdge ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Configure the pins as edge sensitive in Pin interrupt block.

**Parameters**

| | |
|---:|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins (ORed value of PININTCH∗) |

**Returns**

Nothing

Definition at line 169 of file pinint_5411x.h.

#### 6.36.4.24 __STATIC_INLINE void Chip_PININT_SetPinModeLevel ( LPC_PIN_INT_T ∗ *pPININT,* uint32_t *pins* )

Configure the pins as level sensitive in Pin interrupt block.

**Parameters**

| | |
|---:|---|
| *pPININT* | : The base address of Pin interrupt block |
| *pins* | : Pins (ORed value of PININTCH∗) |

**Returns**

Nothing

Definition at line 180 of file pinint_5411x.h.

## 6.37 CHIP: LPC5411X Power LIBRARY functions

### 6.37.1 Detailed Description

The power library provides functions to control system power usage and place the device into low power modes.

**Clock shutdown in sleep and power down modes**
When using the Chip_POWER_EnterPowerMode() function, system clocks are shutdown based on the selected sleep or power down mode and the device version being used. The following list details which clocks are shut down in which modes for which device versions. You can keep a clock enabled for a sleep or power down mode by enabling it in the 'peripheral_ctrl' field in the Chip_POWER_EnterPowerMode() function.

Mode: Sleep
No clocks are disabled for any chip version.

Mode: Deep sleep
SYSCON_PDRUNCFG_PD_IRC_OSC
SYSCON_PDRUNCFG_PD_IRC
SYSCON_PDRUNCFG_PD_FLASH (v17.1 and later only)
SYSCON_PDRUNCFG_PD_BOD_INTR
SYSCON_PDRUNCFG_PD_ADC0
SYSCON_PDRUNCFG_PD_ROM
SYSCON_PDRUNCFG_PD_VDDA_ENA
SYSCON_PDRUNCFG_PD_SYS_PLL
SYSCON_PDRUNCFG_PD_VREFP

Mode: Power down
SYSCON_PDRUNCFG_PD_IRC_OSC
SYSCON_PDRUNCFG_PD_IRC
SYSCON_PDRUNCFG_PD_FLASH (v17.1 and later only)
SYSCON_PDRUNCFG_PD_BOD_RST
SYSCON_PDRUNCFG_PD_BOD_INTR
SYSCON_PDRUNCFG_PD_ADC0
SYSCON_PDRUNCFG_PD_SRAM0B
SYSCON_PDRUNCFG_PD_SRAM1
SYSCON_PDRUNCFG_PD_SRAM2
SYSCON_PDRUNCFG_PD_ROM
SYSCON_PDRUNCFG_PD_VDDA_ENA
SYSCON_PDRUNCFG_PD_WDT_OSC
SYSCON_PDRUNCFG_PD_SYS_PLL
SYSCON_PDRUNCFG_PD_VREFP
SYSCON_PDRUNCFG_PD_32K_OSC

Mode: Deep power down
All clocks are disabled for all chip versions.

If you are using a peripheral was a wakeup source for a power down mode, it needs to be kept active with the call to Chip_POWER_EnterPowerMode(). For example, if you are using the RTC to wake the system up from power down mode, the 32KHz RTC oscillator needs to remain active, so the power down call would look like this:
Chip_POWER_EnterPowerMode(POWER_POWER_DOWN, SYSCON_PDRUNCFG_PD_32K_OSC);
If your application uses internal RAM beyond the first 8K, you will also need to prevent power down of the IRAM like this:
Chip_POWER_EnterPowerMode(POWER_POWER_DOWN, (SYSCON_PDRUNCFG_PD_32K_OSC | SYSCON_PDRUNCFG_PD_SRAM0A));

## Macros

- #define LPC5411X_ROMVER_0 (0x1100)
- #define LPC5411X_ROMVER_1 (0x1101)
- #define LPC5411X_ROMVER_2 (0x1102)

## Enumerations

- enum POWER_MODE_T { POWER_SLEEP = 0, POWER_DEEP_SLEEP, POWER_DEEP_POWER_DO↩
WN }

## Functions

- void Chip_POWER_SetFROHFRate (uint32_t freq)

    *Sets the High Frequency FRO rate to (48MHz or 96MHz)*
- uint32_t Chip_POWER_SetPLL (uint32_t multiply_by, uint32_t input_freq)

    *Sets up the System PLL given the PLL input frequency and feedback multiplier.*
- uint32_t Chip_POWER_SetVoltage (uint32_t desired_freq)

    *Set optimal system voltage based on passed system frequency.*
- void Chip_POWER_SetLowPowerVoltage (uint32_t freq)

    *Set low-power voltage levels for LP mode.*
- void Chip_POWER_EnterPowerMode (POWER_MODE_T mode, uint32_t peripheral_ctrl)

    *Enters the selected power state.*
- uint32_t Chip_POWER_GetROMVersion (void)

    *Return ROM version.*

## 6.37.2 Macro Definition Documentation

### 6.37.2.1 #define LPC5411X_ROMVER_0 (0x1100)

Definition at line 152 of file power_lib_5411x.h.

### 6.37.2.2 #define LPC5411X_ROMVER_1 (0x1101)

Definition at line 153 of file power_lib_5411x.h.

### 6.37.2.3 #define LPC5411X_ROMVER_2 (0x1102)

Definition at line 154 of file power_lib_5411x.h.

## 6.37.3 Enumeration Type Documentation

### 6.37.3.1 enum POWER_MODE_T

Enumerator

***POWER_SLEEP***

***POWER_DEEP_SLEEP***

***POWER_DEEP_POWER_DOWN***

Definition at line 100 of file power_lib_5411x.h.

## 6.37.4 Function Documentation

### 6.37.4.1 void Chip_POWER_EnterPowerMode ( POWER_MODE_T *mode*, uint32_t *peripheral_ctrl* )

Enters the selected power state.

**Parameters**

| | |
|---|---|
| *mode* | : Power mode |
| *peripheral_ctrl* | : Peripherals that will remain powered up in the power state |

**Returns**

Nothing

**Note**

The 'peripheral_ctrl' field is a bitmask of bits from the PDRUNCFG register (SYSCON_PDRUNCFG_PD_∗) that describe which peripherals can wake up the chip from the power state. These peripherals are not powered down during the power state.

### 6.37.4.2 uint32_t Chip_POWER_GetROMVersion ( void )

Return ROM version.

**Returns**

ROM version

**Note**

Will return one of the following version numbers:
(0x1100) for v17.0 ROMs.
(0x1101) for v17.1 ROMs.
(0x1102) for v17.2 ROMs.

### 6.37.4.3 void Chip_POWER_SetFROHFRate ( uint32_t *freq* )

Sets the High Frequency FRO rate to (48MHz or 96MHz)

**Parameters**

| | |
|---|---|
| *freq* | : Frequency selection for FRO |

**Returns**

none

### 6.37.4.4 void Chip_POWER_SetLowPowerVoltage ( uint32_t *freq* )

Set low-power voltage levels for LP mode.

**Parameters**

| | |
|---|---|
| *frequency* | : This is the frequency at which CPU is running. |

**Note**

Low power mode is only possible at 12MHz and 48MHz FRO

**Returns**

Nothing

#### 6.37.4.5 uint32_t Chip_POWER_SetPLL ( uint32_t *multiply_by,* uint32_t *input_freq* )

Sets up the System PLL given the PLL input frequency and feedback multiplier.

**Parameters**

| | |
|---|---|
| *multiply_by* | : PLL multiplier, minimum of 1, maximum of 16 |
| *input_freq* | : Input frequency into the PLL |

**Returns**

LPC_OK on success, or an error code (see error.h)

#### 6.37.4.6 uint32_t Chip_POWER_SetVoltage ( uint32_t *desired_freq* )

Set optimal system voltage based on passed system frequency.

**Parameters**

| | |
|---|---|
| *desired_freq* | : System (CPU) frequency |

**Returns**

LPC_OK on success, or an error code (see error.h)

**Note**

This function will adjust the system voltages to the lowest levels that will support the passed CPU frequency.

# 6.38 CHIP: LPC5411X Power Management declarations and functions

## 6.38.1 Detailed Description

### Data Structures

- struct LPC_PMU_T

  *PMU register block structure.*

### Macros

- #define PMU_BOD_RST (1 << 6)
- #define PMU_BOD_INT (1 << 7)

### Enumerations

- enum CHIP_PMU_BODRSTLVL_T {
  PMU_BODRSTLVL_0, PMU_BODRSTLVL_1_50V = PMU_BODRSTLVL_0, PMU_BODRSTLVL_1, PMU_BODRSTLVL_1_85V = PMU_BODRSTLVL_1,
  PMU_BODRSTLVL_2, PMU_BODRSTLVL_2_00V = PMU_BODRSTLVL_2, PMU_BODRSTLVL_3, PMU_BODRSTLVL_2_30V = PMU_BODRSTLVL_3 }
- enum CHIP_PMU_BODRINTVAL_T {
  PMU_BODINTVAL_LVL0, PMU_BODINTVAL_2_05v = PMU_BODINTVAL_LVL0, PMU_BODINTVAL_LVL1, PMU_BODINTVAL_2_45v = PMU_BODINTVAL_LVL1,
  PMU_BODINTVAL_LVL2, PMU_BODINTVAL_2_75v = PMU_BODINTVAL_LVL2, PMU_BODINTVAL_LVL3, PMU_BODINTVAL_3_05v = PMU_BODINTVAL_LVL3 }

### Functions

- __STATIC_INLINE void Chip_PMU_SetBODLevels (CHIP_PMU_BODRSTLVL_T rstlvl, CHIP_PMU_BODRINTVAL_T intlvl)

  *Set brown-out detection interrupt and reset levels.*
- __STATIC_INLINE void Chip_PMU_EnableBODReset (void)

  *Enable brown-out detection reset.*
- __STATIC_INLINE void Chip_PMU_DisableBODReset (void)

  *Disable brown-out detection reset.*
- __STATIC_INLINE void Chip_PMU_EnableBODInt (void)

  *Enable brown-out detection interrupt.*
- __STATIC_INLINE void Chip_PMU_DisableBODInt (void)

  *Disable brown-out detection interrupt.*

## 6.38.2 Macro Definition Documentation

### 6.38.2.1 #define PMU_BOD_INT (1 << 7)

brown-out detection interrupt status (in BODCTRL register)

Definition at line 88 of file pmu_5411x.h.

### 6.38.2.2 #define PMU_BOD_RST (1 << 6)

brown-out detection reset status (in BODCTRL register)

Definition at line 84 of file pmu_5411x.h.

## 6.38.3 Enumeration Type Documentation

### 6.38.3.1 enum CHIP_PMU_BODRINTVAL_T

Brown-out detector interrupt level

**Enumerator**

***PMU_BODINTVAL_LVL0*** Brown-out interrupt at ∼2.05v
***PMU_BODINTVAL_2_05v***
***PMU_BODINTVAL_LVL1*** Brown-out interrupt at ∼2.45v
***PMU_BODINTVAL_2_45v***
***PMU_BODINTVAL_LVL2*** Brown-out interrupt at ∼2.75v
***PMU_BODINTVAL_2_75v***
***PMU_BODINTVAL_LVL3*** Brown-out interrupt at ∼3.05v
***PMU_BODINTVAL_3_05v***

Definition at line 70 of file pmu_5411x.h.

### 6.38.3.2 enum CHIP_PMU_BODRSTLVL_T

Brown-out detector reset level

**Enumerator**

***PMU_BODRSTLVL_0*** Brown-out reset at ∼1.5v
***PMU_BODRSTLVL_1_50V***
***PMU_BODRSTLVL_1*** Brown-out reset at ∼1.85v
***PMU_BODRSTLVL_1_85V***
***PMU_BODRSTLVL_2*** Brown-out reset at ∼2.0v
***PMU_BODRSTLVL_2_00V***
***PMU_BODRSTLVL_3*** Brown-out reset at ∼2.3v
***PMU_BODRSTLVL_2_30V***

Definition at line 56 of file pmu_5411x.h.

## 6.38.4 Function Documentation

### 6.38.4.1 __STATIC_INLINE void Chip_PMU_DisableBODInt ( void )

Disable brown-out detection interrupt.

**Returns**

Nothing

Definition at line 135 of file pmu_5411x.h.

### 6.38.4.2 __STATIC_INLINE void Chip_PMU_DisableBODReset ( void )

Disable brown-out detection reset.

**Returns**

Nothing

Definition at line 117 of file pmu_5411x.h.

#### 6.38.4.3 __STATIC_INLINE void Chip_PMU_EnableBODInt ( void )

Enable brown-out detection interrupt.

**Returns**

Nothing

Definition at line 126 of file pmu_5411x.h.

#### 6.38.4.4 __STATIC_INLINE void Chip_PMU_EnableBODReset ( void )

Enable brown-out detection reset.

**Returns**

Nothing

Definition at line 108 of file pmu_5411x.h.

#### 6.38.4.5 __STATIC_INLINE void Chip_PMU_SetBODLevels ( CHIP_PMU_BODRSTLVL_T *rstlvl*, CHIP_PMU_BODRINTVAL_T *intlvl* )

Set brown-out detection interrupt and reset levels.

**Parameters**

| | |
|---|---|
| *rstlvl* | : Brown-out detector reset level |
| *intlvl* | : Brown-out interrupt level |

**Returns**

Nothing

**Note**

Brown-out detection reset will be disabled upon exiting this function. Use Chip_PMU_EnableBODReset() to re-enable.

Definition at line 98 of file pmu_5411x.h.

# 6.39 CHIP: LPC5411X ROM API declarations and functions

## 6.39.1 Detailed Description

### Data Structures

- struct LPC_ROM_API_T

  *High level ROM API structure.*

### Macros

- #define LPC_ROM_API_BASE_LOC 0x03000200UL
- #define LPC_ROM_API (∗(LPC_ROM_API_T ∗ ∗) LPC_ROM_API_BASE_LOC)
- #define IAP_ENTRY_LOCATION 0x03000205

### Functions

- static INLINE void iap_entry (unsigned int cmd_param[5], unsigned int status_result[4])

  *LPC5410x IAP_ENTRY API function type.*

## 6.39.2 Macro Definition Documentation

### 6.39.2.1 #define IAP_ENTRY_LOCATION 0x03000205

Definition at line 78 of file romapi_5411x.h.

### 6.39.2.2 #define LPC_ROM_API (∗(LPC_ROM_API_T ∗ ∗) LPC_ROM_API_BASE_LOC)

Definition at line 72 of file romapi_5411x.h.

### 6.39.2.3 #define LPC_ROM_API_BASE_LOC 0x03000200UL

Definition at line 71 of file romapi_5411x.h.

## 6.39.3 Function Documentation

### 6.39.3.1 static INLINE void iap_entry ( unsigned int *cmd_param[5],* unsigned int *status_result[4]* ) [static]

LPC5410x IAP_ENTRY API function type.

Definition at line 83 of file romapi_5411x.h.

## 6.40 CHIP: LPC5411X Real Time clock

### 6.40.1 Detailed Description

### Data Structures

- struct LPC_RTC_T

  *LPC5411X Real Time clock register block structure.*

### Macros

- #define RTC_CTRL_SWRESET (1 << 0)
- #define RTC_CTRL_ALARM1HZ (1 << 2)
- #define RTC_CTRL_WAKE1KHZ (1 << 3)
- #define RTC_CTRL_ALARMDPD_EN (1 << 4)
- #define RTC_CTRL_WAKEDPD_EN (1 << 5)
- #define RTC_CTRL_RTC1KHZ_EN (1 << 6)
- #define RTC_CTRL_RTC_EN (1 << 7)
- #define RTC_CTRL_RTC_OSC_PD (1 << 8)
- #define RTC_CTRL_RTC_OSC_BYPASS (1 << 9)
- #define RTC_CTRL_MASK ((uint32_t) 0x0000003FD)

### Functions

- __STATIC_INLINE void Chip_RTC_Init (LPC_RTC_T ∗pRTC)

  *Initialize the RTC peripheral.*
- __STATIC_INLINE void Chip_RTC_DeInit (LPC_RTC_T ∗pRTC)

  *De-initialize the RTC peripheral.*
- __STATIC_INLINE void Chip_RTC_EnableOptions (LPC_RTC_T ∗pRTC, uint32_t flags)

  *Enable RTC options.*
- __STATIC_INLINE void Chip_RTC_DisableOptions (LPC_RTC_T ∗pRTC, uint32_t flags)

  *Disable RTC options.*
- __STATIC_INLINE void Chip_RTC_Reset (LPC_RTC_T ∗pRTC)

  *Reset RTC.*
- __STATIC_INLINE void Chip_RTC_Enable (LPC_RTC_T ∗pRTC)

  *Enables the RTC.*
- __STATIC_INLINE void Chip_RTC_Disable (LPC_RTC_T ∗pRTC)

  *Disables the RTC.*
- __STATIC_INLINE void Chip_RTC_PowerUp (LPC_RTC_T ∗pRTC)

  *Power up the RTC.*
- __STATIC_INLINE void Chip_RTC_PowerDown (LPC_RTC_T ∗pRTC)

  *Disables the RTC.*
- __STATIC_INLINE void Chip_RTC_Enable1KHZ (LPC_RTC_T ∗pRTC)

  *Enables the RTC 1KHz high resolution timer.*
- __STATIC_INLINE void Chip_RTC_Disable1KHZ (LPC_RTC_T ∗pRTC)

  *Disables the RTC 1KHz high resolution timer.*
- __STATIC_INLINE void Chip_RTC_EnableWakeup (LPC_RTC_T ∗pRTC, uint32_t ints)

  *Enables selected RTC wakeup events.*
- __STATIC_INLINE void Chip_RTC_DisableWakeup (LPC_RTC_T ∗pRTC, uint32_t ints)

  *Disables selected RTC wakeup events.*
- __STATIC_INLINE void Chip_RTC_ClearStatus (LPC_RTC_T ∗pRTC, uint32_t stsMask)

*Clears latched RTC statuses.*

- __STATIC_INLINE uint32_t Chip_RTC_GetStatus (LPC_RTC_T ∗pRTC)

  *Return RTC control/status register.*
- __STATIC_INLINE void Chip_RTC_SetAlarm (LPC_RTC_T ∗pRTC, uint32_t count)

  *Set RTC match value for alarm status/interrupt.*
- __STATIC_INLINE uint32_t Chip_RTC_GetAlarm (LPC_RTC_T ∗pRTC)

  *Return the RTC match value used for alarm status/interrupt.*
- __STATIC_INLINE void Chip_RTC_SetCount (LPC_RTC_T ∗pRTC, uint32_t count)

  *Set RTC match count for 1 second timer count.*
- __STATIC_INLINE uint32_t Chip_RTC_GetCount (LPC_RTC_T ∗pRTC)

  *Get current RTC 1 second timer count.*
- __STATIC_INLINE void Chip_RTC_SetWake (LPC_RTC_T ∗pRTC, uint16_t count)

  *Set RTC wake count countdown value (in mS ticks)*
- __STATIC_INLINE uint16_t Chip_RTC_GetWake (LPC_RTC_T ∗pRTC)

  *Get RTC wake count countdown value.*

## 6.40.2 Macro Definition Documentation

### 6.40.2.1 #define RTC_CTRL_ALARM1HZ (1 << 2)

RTC 1 Hz timer alarm flag status (match) bit

Definition at line 56 of file rtc_5411x.h.

### 6.40.2.2 #define RTC_CTRL_ALARMDPD_EN (1 << 4)

RTC 1 Hz timer alarm for Deep power-down enable bit

Definition at line 58 of file rtc_5411x.h.

### 6.40.2.3 #define RTC_CTRL_MASK ((uint32_t) 0x0000003FD)

RTC Control register Mask for reserved bits

Definition at line 64 of file rtc_5411x.h.

### 6.40.2.4 #define RTC_CTRL_RTC1KHZ_EN (1 << 6)

RTC 1 kHz clock enable bit

Definition at line 60 of file rtc_5411x.h.

### 6.40.2.5 #define RTC_CTRL_RTC_EN (1 << 7)

RTC enable bit

Definition at line 61 of file rtc_5411x.h.

### 6.40.2.6 #define RTC_CTRL_RTC_OSC_BYPASS (1 << 9)

RTC power-down bit

Definition at line 63 of file rtc_5411x.h.

#### 6.40.2.7 #define RTC_CTRL_RTC_OSC_PD (1 << 8)

RTC power-down bit

Definition at line 62 of file rtc_5411x.h.

#### 6.40.2.8 #define RTC_CTRL_SWRESET (1 << 0)

Apply reset to RTC

Definition at line 55 of file rtc_5411x.h.

#### 6.40.2.9 #define RTC_CTRL_WAKE1KHZ (1 << 3)

RTC 1 kHz timer wake-up flag status (timeout) bit

Definition at line 57 of file rtc_5411x.h.

#### 6.40.2.10 #define RTC_CTRL_WAKEDPD_EN (1 << 5)

RTC 1 kHz timer wake-up for Deep power-down enable bit

Definition at line 59 of file rtc_5411x.h.

### 6.40.3 Function Documentation

#### 6.40.3.1 __STATIC_INLINE void Chip_RTC_ClearStatus ( LPC_RTC_T ∗ *pRTC*, uint32_t *stsMask* )

Clears latched RTC statuses.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |
| *stsMask* | : OR'ed status bits to clear |

**Returns**

Nothing

**Note**

Use and OR'ed stsMask value of RTC_CTRL_ALARM1HZ, and RTC_CTRL_WAKE1KHZ to clear specific RTC states.

Definition at line 240 of file rtc_5411x.h.

#### 6.40.3.2 __STATIC_INLINE void Chip_RTC_DeInit ( LPC_RTC_T ∗ *pRTC* )

De-initialize the RTC peripheral.

**Parameters**

| | |
|---|---|
| *pRTC* | : RTC peripheral selected |

**Returns**

None

Definition at line 82 of file rtc_5411x.h.

#### 6.40.3.3 __STATIC_INLINE void Chip_RTC_Disable ( LPC_RTC_T ∗ *pRTC* )

Disables the RTC.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Nothing

**Note**

You can also use Chip_RTC_DisableOptions() with the RTC_CTRL_RTC_EN flag to enable the RTC.

Definition at line 148 of file rtc_5411x.h.

#### 6.40.3.4 __STATIC_INLINE void Chip_RTC_Disable1KHZ ( LPC_RTC_T ∗ *pRTC* )

Disables the RTC 1KHz high resolution timer.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Nothing

**Note**

You can also use Chip_RTC_DisableOptions() with the RTC_CTRL_RTC1KHZ_EN flag to disable the high resolution timer.

Definition at line 197 of file rtc_5411x.h.

#### 6.40.3.5 __STATIC_INLINE void Chip_RTC_DisableOptions ( LPC_RTC_T ∗ *pRTC,* uint32_t *flags* )

Disable RTC options.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |
| *flags* | : And OR'ed value of RTC_CTRL_∗ definitions to disable |

**Returns**

Nothing

**Note**

You can enable multiple RTC options at once using this function by OR'ing them together. It is recommended to only use the RTC_CTRL_ALARMDPD_EN, RTC_CTRL_WAKEDPD_EN, RTC_CTRL_RTC1KHZ_EN, and RTC_CTRL_RTC_EN flags with this function.

Definition at line 112 of file rtc_5411x.h.

#### 6.40.3.6 __STATIC_INLINE void Chip_RTC_DisableWakeup ( LPC_RTC_T ∗ *pRTC,* uint32_t *ints* )

Disables selected RTC wakeup events.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |
| *ints* | : Wakeup events to disable |

**Returns**

Nothing

**Note**

Select either one or both (OR'ed) RTC_CTRL_ALARMDPD_EN and RTC_CTRL_WAKEDPD_EN values to disabled. You can also use Chip_RTC_DisableOptions() with the flags to disable the events.

Definition at line 227 of file rtc_5411x.h.

#### 6.40.3.7 __STATIC_INLINE void Chip_RTC_Enable ( LPC_RTC_T ∗ *pRTC* )

Enables the RTC.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Nothing

**Note**

You can also use Chip_RTC_EnableOptions() with the RTC_CTRL_RTC_EN flag to enable the RTC.

Definition at line 136 of file rtc_5411x.h.

#### 6.40.3.8 __STATIC_INLINE void Chip_RTC_Enable1KHZ ( LPC_RTC_T ∗ *pRTC* )

Enables the RTC 1KHz high resolution timer.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Nothing

**Note**

You can also use Chip_RTC_EnableOptions() with the RTC_CTRL_RTC1KHZ_EN flag to enable the high resolution timer.

Definition at line 184 of file rtc_5411x.h.

#### 6.40.3.9 __STATIC_INLINE void Chip_RTC_EnableOptions ( LPC_RTC_T ∗ *pRTC,* uint32_t *flags* )

Enable RTC options.

**Parameters**

| | |
|---:|---|
| *pRTC* | : The base address of RTC block |
| *flags* | : And OR'ed value of RTC_CTRL_∗ definitions to enable |

**Returns**

Nothing

**Note**

You can enable multiple RTC options at once using this function by OR'ing them together. It is recommended to only use the RTC_CTRL_ALARMDPD_EN, RTC_CTRL_WAKEDPD_EN, RTC_CTRL_RTC1KHZ_EN, and RTC_CTRL_RTC_EN flags with this function.

Definition at line 97 of file rtc_5411x.h.

#### 6.40.3.10 __STATIC_INLINE void Chip_RTC_EnableWakeup ( LPC_RTC_T ∗ *pRTC*, uint32_t *ints* )

Enables selected RTC wakeup events.

**Parameters**

| | |
|---:|---|
| *pRTC* | : The base address of RTC block |
| *ints* | : Wakeup events to enable |

**Returns**

Nothing

**Note**

Select either one or both (OR'ed) RTC_CTRL_ALARMDPD_EN and RTC_CTRL_WAKEDPD_EN values to enabled. You can also use Chip_RTC_EnableOptions() with the flags to enable the events.

Definition at line 212 of file rtc_5411x.h.

#### 6.40.3.11 __STATIC_INLINE uint32_t Chip_RTC_GetAlarm ( LPC_RTC_T ∗ *pRTC* )

Return the RTC match value used for alarm status/interrupt.

**Parameters**

| | |
|---:|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Alarm event time

Definition at line 274 of file rtc_5411x.h.

#### 6.40.3.12 __STATIC_INLINE uint32_t Chip_RTC_GetCount ( LPC_RTC_T ∗ *pRTC* )

Get current RTC 1 second timer count.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

current RTC 1 second timer count

Definition at line 298 of file rtc_5411x.h.

#### 6.40.3.13 __STATIC_INLINE uint32_t Chip_RTC_GetStatus ( LPC_RTC_T ∗ *pRTC* )

Return RTC control/status register.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

The current RTC control/status register

**Note**

Mask the return value with a RTC_CTRL_∗ definitions to determine which bits are set. For example, mask the return value with RTC_CTRL_ALARM1HZ to determine if the alarm interrupt is pending.

Definition at line 253 of file rtc_5411x.h.

#### 6.40.3.14 __STATIC_INLINE uint16_t Chip_RTC_GetWake ( LPC_RTC_T ∗ *pRTC* )

Get RTC wake count countdown value.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

current RTC wake count countdown value (in mS)

Definition at line 321 of file rtc_5411x.h.

#### 6.40.3.15 __STATIC_INLINE void Chip_RTC_Init ( LPC_RTC_T ∗ *pRTC* )

Initialize the RTC peripheral.

**Parameters**

| | |
|---|---|
| *pRTC* | : RTC peripheral selected |

**Returns**

None

Definition at line 72 of file rtc_5411x.h.

#### 6.40.3.16 __STATIC_INLINE void Chip_RTC_PowerDown ( LPC_RTC_T ∗ *pRTC* )

Disables the RTC.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Nothing

**Note**

You can also use Chip_RTC_DisableOptions() with the RTC_CTRL_RTC_EN flag to enable the RTC.

Definition at line 170 of file rtc_5411x.h.

#### 6.40.3.17 __STATIC_INLINE void Chip_RTC_PowerUp ( LPC_RTC_T ∗ *pRTC* )

Power up the RTC.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Nothing

Definition at line 158 of file rtc_5411x.h.

#### 6.40.3.18 __STATIC_INLINE void Chip_RTC_Reset ( LPC_RTC_T ∗ *pRTC* )

Reset RTC.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

**Returns**

Nothing

**Note**

The RTC state will be returned to it's default.

Definition at line 123 of file rtc_5411x.h.

#### 6.40.3.19 __STATIC_INLINE void Chip_RTC_SetAlarm ( LPC_RTC_T ∗ *pRTC,* uint32_t *count* )

Set RTC match value for alarm status/interrupt.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |

| | |
|---|---|
| *count* | : Alarm event time |

**Returns**

Nothing

Definition at line 264 of file rtc_5411x.h.

#### 6.40.3.20 __STATIC_INLINE void Chip_RTC_SetCount ( LPC_RTC_T ∗ *pRTC,* uint32_t *count* )

Set RTC match count for 1 second timer count.

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |
| *count* | : Initial count to set |

**Returns**

Nothing

**Note**

Only write to this register when the RTC_CTRL_RTC_EN bit in the CTRL Register is 0. The counter increments one second after the RTC_CTRL_RTC_EN bit is set.

Definition at line 288 of file rtc_5411x.h.

#### 6.40.3.21 __STATIC_INLINE void Chip_RTC_SetWake ( LPC_RTC_T ∗ *pRTC,* uint16_t *count* )

Set RTC wake count countdown value (in mS ticks)

**Parameters**

| | |
|---|---|
| *pRTC* | : The base address of RTC block |
| *count* | : wakeup time in milliSeconds |

**Returns**

Nothing

**Note**

A write pre-loads a start count value into the wake-up timer and initializes a count-down sequence.

Definition at line 311 of file rtc_5411x.h.

## 6.41 CHIP: LPC5411X SPI driver

### 6.41.1 Detailed Description

The SPI interface is provided via FlexCOMM module in the CHIP, to associate a SPI to a flexcom the source file must add a define like **#define SPI0_FLEXCOMM 5** which will make LPC_SPI0 use LPC_FLEXCOMM5, interrupt service function name and IRQ number DMA REQUEST number will all be mapped automatically. **Note that this define must be available/visible to all the sources that uses SPI.**

The SPI driver by default will use FIFOs for TX and RX. TX and RX FIFOs has a depth of 8 entries each.

#### Data Structures

- struct LPC_SPI_T

  *SPI register block structure.*
- struct SPI_CFGSETUP_T

#### Modules

- CHIP: LPC5411X SPI master driver
- CHIP: LPC5411X SPI slave driver

#### Macros

- #define SPI_CFG_BITMASK (0x0FBD) /∗∗ SPI register bit mask ∗/
- #define SPI_CFG_SPI_EN (1 << 0) /∗∗ SPI Slave Mode Select ∗/
- #define SPI_CFG_SLAVE_EN (0 << 0) /∗∗ SPI Master Mode Select ∗/
- #define SPI_CFG_MASTER_EN (1 << 2) /∗∗ SPI MSB First mode enable ∗/
- #define SPI_CFG_MSB_FIRST_EN (0 << 3) /∗∗ SPI LSB First mode enable ∗/
- #define SPI_CFG_LSB_FIRST_EN (1 << 3) /∗∗ SPI Clock Phase Select ∗/
- #define SPI_CFG_CPHA_FIRST (0 << 4) /∗∗ Capture data on the first edge, Change data on the following edge ∗/
- #define SPI_CFG_CPHA_SECOND (1 << 4) /∗∗ SPI Clock Polarity Select ∗/
- #define SPI_CFG_CPOL_LO (0 << 5) /∗∗ The rest state of the clock (between frames) is low. ∗/
- #define SPI_CFG_CPOL_HI (1 << 5) /∗∗ The rest state of the clock (between frames) is high. ∗/
- #define SPI_CFG_LBM_EN (1 << 7) /∗∗ SPI control 1 loopback mode enable ∗/
- #define SPI_CFG_SPOL_LO (0 << 8) /∗∗ SPI SSEL0 Polarity Select ∗/
- #define SPI_CFG_SPOL_HI (1 << 8) /∗∗ SSEL0 is active High ∗/
- #define SPI_CFG_SPOLNUM_HI(n) (1 << ((n) + 8)) /∗∗ SSELN is active High, selects 0 - 3 ∗/
- #define SPI_DLY_BITMASK (0xFFFF) /∗∗ SPI DLY Register Mask ∗/
- #define SPI_DLY_PRE_DELAY(n) (((n) & 0x0F) << 0) /∗∗ Time in SPI clocks between SSEL assertion and the beginning of a data frame ∗/
- #define SPI_DLY_POST_DELAY(n) (((n) & 0x0F) << 4) /∗∗ Time in SPI clocks between the end of a data frame and SSEL deassertion. ∗/
- #define SPI_DLY_FRAME_DELAY(n) (((n) & 0x0F) << 8) /∗∗ Minimum time in SPI clocks between adjacent data frames. ∗/
- #define SPI_DLY_TRANSFER_DELAY(n) (((n) & 0x0F) << 12) /∗∗ Minimum time in SPI clocks that the SSEL is deasserted between transfers. ∗/
- #define SPI_STAT_BITMASK (0x01F0) /∗∗ SPI STAT Register BitMask ∗/
- #define SPI_STAT_SSA (1 << 4) /∗∗ Slave Select Assert ∗/
- #define SPI_STAT_SSD (1 << 5) /∗∗ Slave Select Deassert ∗/
- #define SPI_STAT_STALLED (1 << 6) /∗∗ Stalled status flag ∗/
- #define SPI_STAT_EOT (1 << 7) /∗∗ End Transfer flag ∗/

- #define SPI_STAT_MSTIDLE (1 << 8) /** Idle status flag */
- #define SPI_INT_BITMASK (0x0130) /** SPI interrupt Enable/Disable bits */
- #define SPI_INT_SSAEN (1 << 4) /** Slave Select is asserted interrupt [BIT-4 of INTENSET/INTENCLR/INTSTAT register] */
- #define SPI_INT_SSDEN (1 << 5) /** Slave Select is deasserted interrupt [BIT-5 of INTENSET/INTENCLR/INTSTAT register] */
- #define SPI_INT_MSTIDLE (1 << 8) /** SPI master is Idle [BIT-8 of INTENSET/INTENCLR/INTSTAT register] */
- #define SPI_FIFOCFG_ENABLETX (1 << 0)

  *SPI FIFO Configuration register bits.*
- #define SPI_FIFOCFG_ENABLERX (1 << 1)
- #define SPI_FIFOCFG_DMATX (1 << 12)
- #define SPI_FIFOCFG_DMARX (1 << 13)
- #define SPI_FIFOCFG_WAKETX (1 << 14)
- #define SPI_FIFOCFG_WAKERX (1 << 15)
- #define SPI_FIFOCFG_EMPTYTX (1 << 16)
- #define SPI_FIFOCFG_EMPTYRX (1 << 17)
- #define SPI_FIFO_DEPTH (8) /** SPI-FIFO How many entries are in the FIFO */

  *Macro defines for FIFO Status register.*
- #define SPI_FIFOSTAT_BITMASK (0x1F1FFB) /** SPI-FIFO STAT Register BitMask */
- #define SPI_FIFOSTAT_TXERR (1 << 0) /** SPI-FIFO transmit error */
- #define SPI_FIFOSTAT_RXERR (1 << 1) /** SPI-FIFO receive error */
- #define SPI_FIFOSTAT_PERINT (1 << 3) /** SPI-FIFO peripheral (SPI) interrupt */
- #define SPI_FIFOSTAT_TXEMPTY (1 << 4) /** SPI-FIFO transmitter empty */
- #define SPI_FIFOSTAT_TXNOTFULL (1 << 5) /** SPI-FIFO transmitter not full */
- #define SPI_FIFOSTAT_RXNOTEMPTY (1 << 6) /** SPI-FIFO receiver not empty */
- #define SPI_FIFOSTAT_RXFULL (1 << 7) /** SPI-FIFO receiver not full */
- #define SPI_FIFOSTAT_TXLVL(val) (((val) >> 8) & 0x1F) /** SPI-FIFO extract transmit level */
- #define SPI_FIFOSTAT_RXLVL(val) (((val) >> 16) & 0x1F) /** SPI-FIFO extract receive level */
- #define SPI_FIFOTRIG_BITMASK (0x000f0f03) /** SPI FIFO trigger settings Register BitMask */

  *UART FIFO trigger settings register defines.*
- #define SPI_FIFOTRIG_TXLVLENA (1 << 0)
- #define SPI_FIFOTRIG_RXLVLENA (1 << 1)
- #define SPI_FIFOTRIG_TXLVL(lvl) ((lvl & 0x0f) << 8)
- #define SPI_FIFOTRIG_RXLVL(lvl) ((lvl & 0x0f) << 16)
- #define SPI_FIFOTRIG_TXLVL_DEFAULT 4
- #define SPI_FIFOTRIG_RXLVL_DEFAULT 0
- #define SPI_FIFOINT_BITMASK (0x001F) /** FIFO interrupt Bit mask */

  *Macro defines for SPI Interrupt enable/disable/status [INTSET/INTCLR/INTSTAT and FIFOINTSET/FIFOINTCLR/FIFOINTSTAT registers].*
- #define SPI_FIFOINT_TXERR (1 << 0) /** TX error interrupt [BIT-0 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */
- #define SPI_FIFOINT_RXERR (1 << 1) /** RX error interrupt [BIT-1 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */
- #define SPI_FIFOINT_TXLVL (1 << 2) /** TX FIFO ready interrupt [BIT-2 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */
- #define SPI_FIFOINT_RXLVL (1 << 3) /** RX Data ready interrupt [BIT-3 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */
- #define SPI_FIFOINT_PERINT (1 << 4) /** SPI peripheral interrupt [BIT-4 of FIFOINTSTAT register] */
- #define SPI_RXDAT_BITMASK (0x1FFFFF) /** SPI RXDAT Register BitMask */
- #define SPI_RXDAT_DATA(n) ((n) & 0xFFFF) /** Receiver Data */
- #define SPI_RXDAT_RXSSELN(n) ((~((n >> 16) & 0x0f)) & 0x0f) /** Determine the active SSEL pin */
- #define SPI_RXDAT_RXSSELN_ACTIVE (0 << 16) /** The state of SSEL pin is active */
- #define SPI_RXDAT_SOT (1 << 20) /** Start of Transfer flag */

- #define SPI_TXDAT_BITMASK (0xF7FFFFF) /∗∗ SPI TXDATCTL Register BitMask ∗/
- #define SPI_TXDAT_DATA(n) ((n) & 0xFFFF) /∗∗ SPI Transmit Data ∗/
- #define SPI_TXDAT_DATA(n) ((n) & 0xFFFF) /∗∗ SPI Transmit Data ∗/
- #define SPI_TXDAT_CTRLMASK (0xF7F) /∗∗ SPI TXDATCTL Register BitMask for control bits only ∗/
- #define SPI_TXDAT_ASSERT_SSEL (0) /∗∗ Assert SSEL0 pin ∗/
- #define SPI_TXDAT_ASSERTNUM_SSEL(n) ((∼(1 << (n))) & 0x0f) /∗∗ Assert SSELN pin ∗/
- #define SPI_TXDAT_DEASSERT_SSEL (1) /∗∗ Deassert SSEL0 pin ∗/
- #define SPI_TXDAT_DEASSERTNUM_SSEL(n) (1 << (n)) /∗∗ Deassert SSELN pin ∗/
- #define SPI_TXDAT_DEASSERT_ALL (0xF) /∗∗ Deassert all SSEL pins ∗/
- #define SPI_TXDAT_EOT (1 << 4) /∗∗ End of Transfer flag (TRANSFER_DELAY is applied after sending the current frame) ∗/
- #define SPI_TXDAT_EOF (1 << 5) /∗∗ End of Frame flag (FRAME_DELAY is applied after sending the current part) ∗/
- #define SPI_TXDAT_RXIGNORE (1 << 6) /∗∗ Receive Ignore Flag ∗/
- #define SPI_TXDAT_FLEN(n) (((n) & 0x0F) << 8) /∗∗ Frame length - 1 ∗/
- #define SPI_TXDAT_FLENMASK (0xF << 8) /∗∗ Frame length mask ∗/
- #define SPI_DIV_VAL(n) ((n) & 0xFFFF) /∗∗ Rate divider value mask (In Master Mode only)∗/
- #define Chip_SPI_ReadFIFO Chip_SPI_ReadRawRXFifo
- #define Chip_SPI_ReadFIFOdata Chip_SPI_ReadRXData
- #define Chip_SPI_WriteFIFOcd Chip_SPI_SetTXCTRLData
- #define Chip_SPI_WriteFIFOdata Chip_SPI_WriteTXData
- #define Chip_SPI_FlushFIFOs Chip_SPI_FlushFifos

## Enumerations

- enum ROM_SPI_CLOCK_MODE_T {
ROM_SPI_CLOCK_CPHA0_CPOL0 = 0, ROM_SPI_CLOCK_MODE0 = ROM_SPI_CLOCK_CPHA0_CPOL0, ROM_SPI_CLOCK_CPHA1_CPOL0 = 1, ROM_SPI_CLOCK_MODE1 = ROM_SPI_CLOCK_CPHA1_CPOL0,
ROM_SPI_CLOCK_CPHA0_CPOL1 = 2, ROM_SPI_CLOCK_MODE2 = ROM_SPI_CLOCK_CPHA0_CPOL1, ROM_SPI_CLOCK_CPHA1_CPOL1 = 3, ROM_SPI_CLOCK_MODE3 = ROM_SPI_CLOCK_CPHA1_CPOL1 }

  *SPI Clock Mode.*
- enum SPI_CLOCK_MODE_T {
SPI_CLOCK_CPHA0_CPOL0 = SPI_CFG_CPOL_LO | SPI_CFG_CPHA_FIRST, SPI_CLOCK_MODE0 = SPI_CLOCK_CPHA0_CPOL0, SPI_CLOCK_CPHA1_CPOL0 = SPI_CFG_CPOL_LO | SPI_CFG_CPHA_SECOND, SPI_CLOCK_MODE1 = SPI_CLOCK_CPHA1_CPOL0,
SPI_CLOCK_CPHA0_CPOL1 = SPI_CFG_CPOL_HI | SPI_CFG_CPHA_FIRST, SPI_CLOCK_MODE2 = SPI_CLOCK_CPHA0_CPOL1, SPI_CLOCK_CPHA1_CPOL1 = SPI_CFG_CPOL_HI | SPI_CFG_CPHA_SECOND, SPI_CLOCK_MODE3 = SPI_CLOCK_CPHA1_CPOL1 }

  *SPI Clock Mode.*

## Functions

- int Chip_SPI_Init (LPC_SPI_T ∗pSPI)

  *Initialize the SPI.*
- __STATIC_INLINE void Chip_SPI_DeInit (LPC_SPI_T ∗pSPI)

  *Disable SPI operation.*
- __STATIC_INLINE void Chip_SPI_SetCFGRegBits (LPC_SPI_T ∗pSPI, uint32_t bits)

  *Set SPI CFG register values.*
- __STATIC_INLINE void Chip_SPI_ClearCFGRegBits (LPC_SPI_T ∗pSPI, uint32_t bits)

  *Clear SPI CFG register values.*
- __STATIC_INLINE void Chip_SPI_Enable (LPC_SPI_T ∗pSPI)

*Enable SPI peripheral.*

- __STATIC_INLINE void Chip_SPI_Disable (LPC_SPI_T ∗pSPI)

  *Disable SPI peripheral.*
- __STATIC_INLINE void Chip_SPI_EnableSlaveMode (LPC_SPI_T ∗pSPI)

  *Enable SPI slave mode.*
- __STATIC_INLINE void Chip_SPI_EnableLSBFirst (LPC_SPI_T ∗pSPI)

  *Enable LSB First transfers.*
- __STATIC_INLINE void Chip_SPI_EnableMSBFirst (LPC_SPI_T ∗pSPI)

  *Enable MSB First transfers.*
- __STATIC_INLINE void Chip_SPI_SetSPIMode (LPC_SPI_T ∗pSPI, SPI_CLOCK_MODE_T mode)

  *Set SPI mode.*
- __STATIC_INLINE void Chip_SPI_SetCSPolHigh (LPC_SPI_T ∗pSPI, uint8_t csNum)

  *Set polarity on the SPI chip select high.*
- __STATIC_INLINE void Chip_SPI_SetCSPolLow (LPC_SPI_T ∗pSPI, uint8_t csNum)

  *Set polarity on the SPI chip select low.*
- void Chip_SPI_ConfigureSPI (LPC_SPI_T ∗pSPI, SPI_CFGSETUP_T ∗pCFG)

  *Setup SPI configuration.*
- __STATIC_INLINE uint32_t Chip_SPI_GetStatus (LPC_SPI_T ∗pSPI)

  *Get the current status of SPI controller.*
- __STATIC_INLINE void Chip_SPI_ClearStatus (LPC_SPI_T ∗pSPI, uint32_t Flag)

  *Clear SPI status.*
- __STATIC_INLINE void Chip_SPI_EnableInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Enable a SPI interrupt.*
- __STATIC_INLINE void Chip_SPI_DisableInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Disable a SPI interrupt.*
- __STATIC_INLINE uint32_t Chip_SPI_GetEnabledInts (LPC_SPI_T ∗pSPI)

  *Return enabled SPI interrupts.*
- __STATIC_INLINE uint32_t Chip_SPI_GetPendingInts (LPC_SPI_T ∗pSPI)

  *Return pending SPI interrupts.*
- __STATIC_INLINE void Chip_SPI_SetFIFOCfg (LPC_SPI_T ∗pSPI, uint32_t cfg)

  *Set FIFO Configuration register.*
- __STATIC_INLINE void Chip_SPI_ClearFIFOCfg (LPC_SPI_T ∗pSPI, uint32_t cfg)

  *Clear FIFO Configuration register.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOStatus (LPC_SPI_T ∗pSPI)

  *Get the current status of SPI controller FIFO.*
- __STATIC_INLINE void Chip_SPI_ClearFIFOStatus (LPC_SPI_T ∗pSPI, uint32_t mask)

  *Clear the FIFO status register.*
- __STATIC_INLINE void Chip_SPI_SetFIFOTrigLevel (LPC_SPI_T ∗pSPI, uint8_t tx_lvl, uint8_t rx_lvl)

  *Setup SPI FIFO trigger-level.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOTrigLevel (LPC_SPI_T ∗pSPI)

  *Get SPI FIFO trigger-level.*
- __STATIC_INLINE void Chip_SPI_EnableFIFOInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Enable a SPI FIFO interrupt.*
- __STATIC_INLINE void Chip_SPI_DisableFIFOInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Disable a SPI FIFO interrupt.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOEnabledInts (LPC_SPI_T ∗pSPI)

  *Return enabled SPI FIFO interrupts.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOPendingInts (LPC_SPI_T ∗pSPI)

  *Return pending SPI FIFO interrupts.*
- __STATIC_INLINE uint32_t Chip_SPI_ReadRawRXFifo (LPC_SPI_T ∗pSPI)

  *Read raw data from receive FIFO with status bits.*

- __STATIC_INLINE uint16_t Chip_SPI_ReadRXData (LPC_SPI_T ∗pSPI)

  *Read data from receive FIFO masking off status bits.*
- __STATIC_INLINE void Chip_SPI_WriteFIFO (LPC_SPI_T ∗pSPI, uint32_t data)

  *Write FIFOWR register: writes 32-bit value to FIFO.*
- __STATIC_INLINE void Chip_SPI_SetTXCTRLData (LPC_SPI_T ∗pSPI, uint16_t ctrl, uint16_t data)

  *Write FIFOWR register: writes control options and data.*
- __STATIC_INLINE void Chip_SPI_WriteTXData (LPC_SPI_T ∗pSPI, uint16_t data)

  *Write data to transmit FIFO.*
- __STATIC_INLINE void Chip_SPI_FlushFifos (LPC_SPI_T ∗pSPI)

  *Flush FIFOs.*

## 6.41.2 Macro Definition Documentation

### 6.41.2.1 #define Chip_SPI_FlushFIFOs Chip_SPI_FlushFifos

Definition at line 676 of file spi_common_5411x.h.

### 6.41.2.2 #define Chip_SPI_ReadFIFO Chip_SPI_ReadRawRXFifo

Definition at line 611 of file spi_common_5411x.h.

### 6.41.2.3 #define Chip_SPI_ReadFIFOdata Chip_SPI_ReadRXData

Definition at line 625 of file spi_common_5411x.h.

### 6.41.2.4 #define Chip_SPI_WriteFIFOcd Chip_SPI_SetTXCTRLData

Definition at line 652 of file spi_common_5411x.h.

### 6.41.2.5 #define Chip_SPI_WriteFIFOdata Chip_SPI_WriteTXData

Definition at line 665 of file spi_common_5411x.h.

### 6.41.2.6 #define SPI_CFG_BITMASK (0x0FBD) /∗∗ SPI register bit mask ∗/

Macro defines for SPI Configuration register

Definition at line 92 of file spi_common_5411x.h.

### 6.41.2.7 #define SPI_CFG_CPHA_FIRST (0 << 4) /∗∗ Capture data on the first edge, Change data on the following edge ∗/

Definition at line 98 of file spi_common_5411x.h.

### 6.41.2.8 #define SPI_CFG_CPHA_SECOND (1 << 4) /∗∗ SPI Clock Polarity Select ∗/

Definition at line 99 of file spi_common_5411x.h.

### 6.41.2.9 #define SPI_CFG_CPOL_HI (1 << 5) /∗∗ The rest state of the clock (between frames) is high. ∗/

Definition at line 101 of file spi_common_5411x.h.

#### 6.41.2.10 #define SPI_CFG_CPOL_LO (0 << 5) /** The rest state of the clock (between frames) is low. */

Definition at line 100 of file spi_common_5411x.h.

#### 6.41.2.11 #define SPI_CFG_LBM_EN (1 << 7) /** SPI control 1 loopback mode enable */

Definition at line 102 of file spi_common_5411x.h.

#### 6.41.2.12 #define SPI_CFG_LSB_FIRST_EN (1 << 3) /** SPI Clock Phase Select */

Definition at line 97 of file spi_common_5411x.h.

#### 6.41.2.13 #define SPI_CFG_MASTER_EN (1 << 2) /** SPI MSB First mode enable */

Definition at line 95 of file spi_common_5411x.h.

#### 6.41.2.14 #define SPI_CFG_MSB_FIRST_EN (0 << 3) /** SPI LSB First mode enable */

Definition at line 96 of file spi_common_5411x.h.

#### 6.41.2.15 #define SPI_CFG_SLAVE_EN (0 << 0) /** SPI Master Mode Select */

Definition at line 94 of file spi_common_5411x.h.

#### 6.41.2.16 #define SPI_CFG_SPI_EN (1 << 0) /** SPI Slave Mode Select */

Definition at line 93 of file spi_common_5411x.h.

#### 6.41.2.17 #define SPI_CFG_SPOL_HI (1 << 8) /** SSEL0 is active High */

Definition at line 104 of file spi_common_5411x.h.

#### 6.41.2.18 #define SPI_CFG_SPOL_LO (0 << 8) /** SPI SSEL0 Polarity Select */

Definition at line 103 of file spi_common_5411x.h.

#### 6.41.2.19 #define SPI_CFG_SPOLNUM_HI( *n* ) (1 << ((n) + 8)) /** SSELN is active High, selects 0 - 3 */

Definition at line 105 of file spi_common_5411x.h.

#### 6.41.2.20 #define SPI_DIV_VAL( *n* ) ((n) & 0xFFFF) /** Rate divider value mask (In Master Mode only)*/

Macro defines for SPI Divider register

Definition at line 219 of file spi_common_5411x.h.

#### 6.41.2.21 #define SPI_DLY_BITMASK (0xFFFF) /** SPI DLY Register Mask */

Macro defines for SPI Delay register

Definition at line 110 of file spi_common_5411x.h.

#### 6.41.2.22 #define SPI_DLY_FRAME_DELAY( *n* ) (((n) & 0x0F) $<<$ 8) /$**$ Minimum time in SPI clocks between adjacent data frames. $*$/

Definition at line 113 of file spi_common_5411x.h.

#### 6.41.2.23 #define SPI_DLY_POST_DELAY( *n* ) (((n) & 0x0F) $<<$ 4) /$**$ Time in SPI clocks between the end of a data frame and SSEL deassertion. $*$/

Definition at line 112 of file spi_common_5411x.h.

#### 6.41.2.24 #define SPI_DLY_PRE_DELAY( *n* ) (((n) & 0x0F) $<<$ 0) /$**$ Time in SPI clocks between SSEL assertion and the beginning of a data frame $*$/

Definition at line 111 of file spi_common_5411x.h.

#### 6.41.2.25 #define SPI_DLY_TRANSFER_DELAY( *n* ) (((n) & 0x0F) $<<$ 12) /$**$ Minimum time in SPI clocks that the SSEL is deasserted between transfers. $*$/

Definition at line 114 of file spi_common_5411x.h.

#### 6.41.2.26 #define SPI_FIFO_DEPTH (8) /$**$ SPI-FIFO How many entries are in the FIFO $*$/

Macro defines for FIFO Status register.

Definition at line 151 of file spi_common_5411x.h.

#### 6.41.2.27 #define SPI_FIFOCFG_DMARX (1 $<<$ 13)

Enable DMA RX

Definition at line 142 of file spi_common_5411x.h.

#### 6.41.2.28 #define SPI_FIFOCFG_DMATX (1 $<<$ 12)

Enable DMA TX

Definition at line 141 of file spi_common_5411x.h.

#### 6.41.2.29 #define SPI_FIFOCFG_EMPTYRX (1 $<<$ 17)

Empty the RX FIFO

Definition at line 146 of file spi_common_5411x.h.

#### 6.41.2.30 #define SPI_FIFOCFG_EMPTYTX (1 $<<$ 16)

Empty the TX FIFO

Definition at line 145 of file spi_common_5411x.h.

#### 6.41.2.31 #define SPI_FIFOCFG_ENABLERX (1 $<<$ 1)

Enable RX FIFO

Definition at line 140 of file spi_common_5411x.h.

#### 6.41.2.32 #define SPI_FIFOCFG_ENABLETX (1 << 0)

SPI FIFO Configuration register bits.

Enable TX FIFO

Definition at line 139 of file spi_common_5411x.h.

#### 6.41.2.33 #define SPI_FIFOCFG_WAKERX (1 << 15)

Enable wakeup triggered by RX

Definition at line 144 of file spi_common_5411x.h.

#### 6.41.2.34 #define SPI_FIFOCFG_WAKETX (1 << 14)

Enable wakeup triggered by TX

Definition at line 143 of file spi_common_5411x.h.

#### 6.41.2.35 #define SPI_FIFOINT_BITMASK (0x001F) /** FIFO interrupt Bit mask */

Macro defines for SPI Interrupt enable/disable/status [INTSET/INTCLR/INTSTAT and FIFOINTSET/FIFOINTCLR/FIFOINTSTAT registers].

Definition at line 178 of file spi_common_5411x.h.

#### 6.41.2.36 #define SPI_FIFOINT_PERINT (1 << 4) /** SPI peripheral interrupt [BIT-4 of FIFOINTSTAT register] */

Definition at line 183 of file spi_common_5411x.h.

#### 6.41.2.37 #define SPI_FIFOINT_RXERR (1 << 1) /** RX error interrupt [BIT-1 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */

Definition at line 180 of file spi_common_5411x.h.

#### 6.41.2.38 #define SPI_FIFOINT_RXLVL (1 << 3) /** RX Data ready interrupt [BIT-3 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */

Definition at line 182 of file spi_common_5411x.h.

#### 6.41.2.39 #define SPI_FIFOINT_TXERR (1 << 0) /** TX error interrupt [BIT-0 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */

Definition at line 179 of file spi_common_5411x.h.

#### 6.41.2.40 #define SPI_FIFOINT_TXLVL (1 << 2) /** TX FIFO ready interrupt [BIT-2 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */

Definition at line 181 of file spi_common_5411x.h.

#### 6.41.2.41 #define SPI_FIFOSTAT_BITMASK (0x1F1FFB) /** SPI-FIFO STAT Register BitMask */

Definition at line 152 of file spi_common_5411x.h.

#### 6.41.2.42 #define SPI_FIFOSTAT_PERINT (1 $<<$ 3) /$**$ SPI-FIFO peripheral (SPI) interrupt $*$/

Definition at line 155 of file spi_common_5411x.h.

#### 6.41.2.43 #define SPI_FIFOSTAT_RXERR (1 $<<$ 1) /$**$ SPI-FIFO receive error $*$/

Definition at line 154 of file spi_common_5411x.h.

#### 6.41.2.44 #define SPI_FIFOSTAT_RXFULL (1 $<<$ 7) /$**$ SPI-FIFO receiver not full $*$/

Definition at line 159 of file spi_common_5411x.h.

#### 6.41.2.45 #define SPI_FIFOSTAT_RXLVL( *val* ) (((val) $>>$ 16) & 0x1F) /$**$ SPI-FIFO extract receive level $*$/

Definition at line 161 of file spi_common_5411x.h.

#### 6.41.2.46 #define SPI_FIFOSTAT_RXNOTEMPTY (1 $<<$ 6) /$**$ SPI-FIFO receiver not empty $*$/

Definition at line 158 of file spi_common_5411x.h.

#### 6.41.2.47 #define SPI_FIFOSTAT_TXEMPTY (1 $<<$ 4) /$**$ SPI-FIFO transmitter empty $*$/

Definition at line 156 of file spi_common_5411x.h.

#### 6.41.2.48 #define SPI_FIFOSTAT_TXERR (1 $<<$ 0) /$**$ SPI-FIFO transmit error $*$/

Definition at line 153 of file spi_common_5411x.h.

#### 6.41.2.49 #define SPI_FIFOSTAT_TXLVL( *val* ) (((val) $>>$ 8) & 0x1F) /$**$ SPI-FIFO extract transmit level $*$/

Definition at line 160 of file spi_common_5411x.h.

#### 6.41.2.50 #define SPI_FIFOSTAT_TXNOTFULL (1 $<<$ 5) /$**$ SPI-FIFO transmitter not full $*$/

Definition at line 157 of file spi_common_5411x.h.

#### 6.41.2.51 #define SPI_FIFOTRIG_BITMASK (0x000f0f03) /$**$ SPI FIFO trigger settings Register BitMask $*$/

UART FIFO trigger settings register defines.

Definition at line 166 of file spi_common_5411x.h.

#### 6.41.2.52 #define SPI_FIFOTRIG_RXLVL( *lvl* ) ((lvl & 0x0f) $<<$ 16)

Set RX Level trigger

Definition at line 170 of file spi_common_5411x.h.

#### 6.41.2.53 #define SPI_FIFOTRIG_RXLVL_DEFAULT 0

Set RX default trigger level to one item

Definition at line 173 of file spi_common_5411x.h.

#### 6.41.2.54 #define SPI_FIFOTRIG_RXLVLENA (1 << 1)

RX level enable

Definition at line 168 of file spi_common_5411x.h.

#### 6.41.2.55 #define SPI_FIFOTRIG_TXLVL( *lvl* ) ((lvl & 0x0f) << 8)

Set TX Level trigger

Definition at line 169 of file spi_common_5411x.h.

#### 6.41.2.56 #define SPI_FIFOTRIG_TXLVL_DEFAULT 4

Set TX default trigger level to half-full

Definition at line 172 of file spi_common_5411x.h.

#### 6.41.2.57 #define SPI_FIFOTRIG_TXLVLENA (1 << 0)

TX level enable

Definition at line 167 of file spi_common_5411x.h.

#### 6.41.2.58 #define SPI_INT_BITMASK (0x0130) /** SPI interrupt Enable/Disable bits */

Macro defines for SPI interrupt register

Definition at line 129 of file spi_common_5411x.h.

#### 6.41.2.59 #define SPI_INT_MSTIDLE (1 << 8) /** SPI master is Idle [BIT-8 of INTENSET/INTENCLR/INTSTAT register] */

Definition at line 132 of file spi_common_5411x.h.

#### 6.41.2.60 #define SPI_INT_SSAEN (1 << 4) /** Slave Select is asserted interrupt [BIT-4 of INTENSET/INTENCLR/INTSTAT register] */

Definition at line 130 of file spi_common_5411x.h.

#### 6.41.2.61 #define SPI_INT_SSDEN (1 << 5) /** Slave Select is deasserted interrupt [BIT-5 of INTENSET/INTENCLR/INTSTAT register] */

Definition at line 131 of file spi_common_5411x.h.

#### 6.41.2.62 #define SPI_RXDAT_BITMASK (0x1FFFFF) /** SPI RXDAT Register BitMask */

Macro defines for SPI FIFO Receiver Data register

Definition at line 188 of file spi_common_5411x.h.

#### 6.41.2.63 #define SPI_RXDAT_DATA( *n* ) ((n) & 0xFFFF) /∗∗ Receiver Data ∗/

Definition at line 189 of file spi_common_5411x.h.

#### 6.41.2.64 #define SPI_RXDAT_RXSSELN( *n* ) ((∼((n >> 16) & 0x0f)) & 0x0f) /∗∗ Determine the active SSEL pin ∗/

Definition at line 190 of file spi_common_5411x.h.

#### 6.41.2.65 #define SPI_RXDAT_RXSSELN_ACTIVE (0 << 16) /∗∗ The state of SSEL pin is active ∗/

Definition at line 191 of file spi_common_5411x.h.

#### 6.41.2.66 #define SPI_RXDAT_SOT (1 << 20) /∗∗ Start of Transfer flag ∗/

Definition at line 192 of file spi_common_5411x.h.

#### 6.41.2.67 #define SPI_STAT_BITMASK (0x01F0) /∗∗ SPI STAT Register BitMask ∗/

Macro defines for SPI Status register

Definition at line 119 of file spi_common_5411x.h.

#### 6.41.2.68 #define SPI_STAT_EOT (1 << 7) /∗∗ End Transfer flag ∗/

Definition at line 123 of file spi_common_5411x.h.

#### 6.41.2.69 #define SPI_STAT_MSTIDLE (1 << 8) /∗∗ Idle status flag ∗/

Definition at line 124 of file spi_common_5411x.h.

#### 6.41.2.70 #define SPI_STAT_SSA (1 << 4) /∗∗ Slave Select Assert ∗/

Definition at line 120 of file spi_common_5411x.h.

#### 6.41.2.71 #define SPI_STAT_SSD (1 << 5) /∗∗ Slave Select Deassert ∗/

Definition at line 121 of file spi_common_5411x.h.

#### 6.41.2.72 #define SPI_STAT_STALLED (1 << 6) /∗∗ Stalled status flag ∗/

Definition at line 122 of file spi_common_5411x.h.

#### 6.41.2.73 #define SPI_TXDAT_ASSERT_SSEL (0) /∗∗ Assert SSEL0 pin ∗/

Definition at line 200 of file spi_common_5411x.h.

#### 6.41.2.74 #define SPI_TXDAT_ASSERTNUM_SSEL( *n* ) ((∼(1 << (n))) & 0x0f) /∗∗ Assert SSELN pin ∗/

Definition at line 201 of file spi_common_5411x.h.

#### 6.41.2.75 #define SPI_TXDAT_BITMASK (0xF7FFFFF) /∗∗ SPI TXDATCTL Register BitMask ∗/

Macro defines for SPI FIFO Transmitter Data and Control register

Definition at line 197 of file spi_common_5411x.h.

#### 6.41.2.76 #define SPI_TXDAT_CTRLMASK (0xF7F) /∗∗ SPI TXDATCTL Register BitMask for control bits only ∗/

Definition at line 199 of file spi_common_5411x.h.

#### 6.41.2.77 #define SPI_TXDAT_DATA( *n* ) ((n) & 0xFFFF) /∗∗ SPI Transmit Data ∗/

Macro defines for SPI Transmitter Data Register

Definition at line 214 of file spi_common_5411x.h.

#### 6.41.2.78 #define SPI_TXDAT_DATA( *n* ) ((n) & 0xFFFF) /∗∗ SPI Transmit Data ∗/

Macro defines for SPI Transmitter Data Register

Definition at line 214 of file spi_common_5411x.h.

#### 6.41.2.79 #define SPI_TXDAT_DEASSERT_ALL (0xF) /∗∗ Deassert all SSEL pins ∗/

Definition at line 204 of file spi_common_5411x.h.

#### 6.41.2.80 #define SPI_TXDAT_DEASSERT_SSEL (1) /∗∗ Deassert SSEL0 pin ∗/

Definition at line 202 of file spi_common_5411x.h.

#### 6.41.2.81 #define SPI_TXDAT_DEASSERTNUM_SSEL( *n* ) (1 << (n)) /∗∗ Deassert SSELN pin ∗/

Definition at line 203 of file spi_common_5411x.h.

#### 6.41.2.82 #define SPI_TXDAT_EOF (1 << 5) /∗∗ End of Frame flag (FRAME_DELAY is applied after sending the current part) ∗/

Definition at line 206 of file spi_common_5411x.h.

#### 6.41.2.83 #define SPI_TXDAT_EOT (1 << 4) /∗∗ End of Transfer flag (TRANSFER_DELAY is applied after sending the current frame) ∗/

Definition at line 205 of file spi_common_5411x.h.

#### 6.41.2.84 #define SPI_TXDAT_FLEN( *n* ) (((n) & 0x0F) << 8) /∗∗ Frame length - 1 ∗/

Definition at line 208 of file spi_common_5411x.h.

#### 6.41.2.85 #define SPI_TXDAT_FLENMASK (0xF << 8) /∗∗ Frame length mask ∗/

Definition at line 209 of file spi_common_5411x.h.

#### 6.41.2.86 #define SPI_TXDAT_RXIGNORE (1 << 6) /** Receive Ignore Flag */

Definition at line 207 of file spi_common_5411x.h.

### 6.41.3 Enumeration Type Documentation

#### 6.41.3.1 enum ROM_SPI_CLOCK_MODE_T

SPI Clock Mode.

**Enumerator**

***ROM_SPI_CLOCK_CPHA0_CPOL0*** CPHA = 0, CPOL = 0

***ROM_SPI_CLOCK_MODE0*** Alias for CPHA = 0, CPOL = 0

***ROM_SPI_CLOCK_CPHA1_CPOL0*** CPHA = 0, CPOL = 1

***ROM_SPI_CLOCK_MODE1*** Alias for CPHA = 0, CPOL = 1

***ROM_SPI_CLOCK_CPHA0_CPOL1*** CPHA = 1, CPOL = 0

***ROM_SPI_CLOCK_MODE2*** Alias for CPHA = 1, CPOL = 0

***ROM_SPI_CLOCK_CPHA1_CPOL1*** CPHA = 1, CPOL = 1

***ROM_SPI_CLOCK_MODE3*** Alias for CPHA = 1, CPOL = 1

Definition at line 222 of file spi_common_5411x.h.

#### 6.41.3.2 enum SPI_CLOCK_MODE_T

SPI Clock Mode.

**Enumerator**

***SPI_CLOCK_CPHA0_CPOL0*** CPHA = 0, CPOL = 0

***SPI_CLOCK_MODE0*** Alias for CPHA = 0, CPOL = 0

***SPI_CLOCK_CPHA1_CPOL0*** CPHA = 0, CPOL = 1

***SPI_CLOCK_MODE1*** Alias for CPHA = 0, CPOL = 1

***SPI_CLOCK_CPHA0_CPOL1*** CPHA = 1, CPOL = 0

***SPI_CLOCK_MODE2*** Alias for CPHA = 1, CPOL = 0

***SPI_CLOCK_CPHA1_CPOL1*** CPHA = 1, CPOL = 1

***SPI_CLOCK_MODE3*** Alias for CPHA = 1, CPOL = 1

Definition at line 329 of file spi_common_5411x.h.

### 6.41.4 Function Documentation

#### 6.41.4.1 __STATIC_INLINE void Chip_SPI_ClearCFGRegBits ( LPC_SPI_T * *pSPI,* uint32_t *bits* )

Clear SPI CFG register values.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *bits* | : CFG register bits to clear, amd OR'ed value of SPI_CFG_∗ definitions |

**Returns**

Nothing

**Note**

This function safely clears only the selected bits in the SPI CFG register. It can be used to disable multiple bits at once.

Definition at line 272 of file spi_common_5411x.h.

#### 6.41.4.2 __STATIC_INLINE void Chip_SPI_ClearFIFOCfg ( LPC_SPI_T ∗ *pSPI,* uint32_t *cfg* )

Clear FIFO Configuration register.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *cfg* | : Configuration value mask (OR'ed SPI_FIFOCFG_∗ values like SPI_FIFOCFG_ENABLETX) |

**Returns**

Nothing

Definition at line 493 of file spi_common_5411x.h.

#### 6.41.4.3 __STATIC_INLINE void Chip_SPI_ClearFIFOStatus ( LPC_SPI_T ∗ *pSPI,* uint32_t *mask* )

Clear the FIFO status register.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *mask* | : Mask of the status bits that needs to be cleared |

**Returns**

Nothing

Definition at line 519 of file spi_common_5411x.h.

#### 6.41.4.4 __STATIC_INLINE void Chip_SPI_ClearStatus ( LPC_SPI_T ∗ *pSPI,* uint32_t *Flag* )

Clear SPI status.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *Flag* | : Clear Flag (Or-ed bit value of SPI_STAT_∗) |

**Returns**

Nothing

Definition at line 411 of file spi_common_5411x.h.

#### 6.41.4.5 void Chip_SPI_ConfigureSPI ( LPC_SPI_T ∗ *pSPI*, SPI_CFGSETUP_T ∗ *pCFG* )

Setup SPI configuration.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *pCFG* | : Pointer to SPI configuration structure |

**Returns**

Nothing

Definition at line 63 of file spi_common_5411x.c.

#### 6.41.4.6 __STATIC_INLINE void Chip_SPI_DeInit ( LPC_SPI_T ∗ *pSPI* )

Disable SPI operation.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Nothing

**Note**

The SPI controller is disabled.

Definition at line 246 of file spi_common_5411x.h.

#### 6.41.4.7 __STATIC_INLINE void Chip_SPI_Disable ( LPC_SPI_T ∗ *pSPI* )

Disable SPI peripheral.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Nothing

Definition at line 292 of file spi_common_5411x.h.

#### 6.41.4.8 __STATIC_INLINE void Chip_SPI_DisableFIFOInts ( LPC_SPI_T ∗ *pSPI,* uint32_t *intMask* )

Disable a SPI FIFO interrupt.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *intMask* | : Or'ed value of SPI_FIFOINT_∗ values to disable (See SPI_FIFOINT_BITMASK) |

**Returns**

Nothing

Definition at line 574 of file spi_common_5411x.h.

#### 6.41.4.9 __STATIC_INLINE void Chip_SPI_DisableInts ( LPC_SPI_T ∗ *pSPI,* uint32_t *intMask* )

Disable a SPI interrupt.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *intMask* | : Or'ed value of SPI_INT_∗ values to disable (See SPI_INT_BITMASK) |

**Returns**

Nothing

Definition at line 433 of file spi_common_5411x.h.

#### 6.41.4.10 __STATIC_INLINE void Chip_SPI_Enable ( LPC_SPI_T ∗ *pSPI* )

Enable SPI peripheral.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Nothing

Definition at line 282 of file spi_common_5411x.h.

#### 6.41.4.11 __STATIC_INLINE void Chip_SPI_EnableFIFOInts ( LPC_SPI_T ∗ *pSPI,* uint32_t *intMask* )

Enable a SPI FIFO interrupt.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *intMask* | : Or'ed value of SPI_FIFOINT_∗ values to enable |

**Returns**

Nothing

Definition at line 563 of file spi_common_5411x.h.

#### 6.41.4.12 __STATIC_INLINE void Chip_SPI_EnableInts ( LPC_SPI_T ∗ *pSPI,* uint32_t *intMask* )

Enable a SPI interrupt.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *intMask* | : Or'ed value of SPI_INT_∗ values to enable |

**Returns**

Nothing

Definition at line 422 of file spi_common_5411x.h.

#### 6.41.4.13 __STATIC_INLINE void Chip_SPI_EnableLSBFirst ( LPC_SPI_T ∗ *pSPI* )

Enable LSB First transfers.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Nothing

Definition at line 313 of file spi_common_5411x.h.

#### 6.41.4.14 __STATIC_INLINE void Chip_SPI_EnableMSBFirst ( LPC_SPI_T * *pSPI* )

Enable MSB First transfers.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Nothing

Definition at line 323 of file spi_common_5411x.h.

#### 6.41.4.15 __STATIC_INLINE void Chip_SPI_EnableSlaveMode ( LPC_SPI_T * *pSPI* )

Enable SPI slave mode.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Nothing

**Note**

SPI master mode will be disabled with this call.

Definition at line 303 of file spi_common_5411x.h.

#### 6.41.4.16 __STATIC_INLINE void Chip_SPI_FlushFifos ( LPC_SPI_T * *pSPI* )

Flush FIFOs.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Nothing

Definition at line 672 of file spi_common_5411x.h.

#### 6.41.4.17 __STATIC_INLINE uint32_t Chip_SPI_GetEnabledInts ( LPC_SPI_T * *pSPI* )

Return enabled SPI interrupts.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

An Or'ed value of SPI_INT_∗ values

**Note**

Mask the return value with a SPI_INT_∗ value to determine (See SPI_INT_BITMASK) if the interrupt is enabled.
INTSTAT contains enabled interrupts, not pending interrupts.

Definition at line 446 of file spi_common_5411x.h.

#### 6.41.4.18 __STATIC_INLINE uint32_t Chip_SPI_GetFIFOEnabledInts ( LPC_SPI_T ∗ *pSPI* )

Return enabled SPI FIFO interrupts.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

An Or'ed value of SPI_FIFOINT_∗ values

Definition at line 584 of file spi_common_5411x.h.

#### 6.41.4.19 __STATIC_INLINE uint32_t Chip_SPI_GetFIFOPendingInts ( LPC_SPI_T ∗ *pSPI* )

Return pending SPI FIFO interrupts.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

An Or'ed value of SPI_FIFOINT_∗ values

Definition at line 594 of file spi_common_5411x.h.

#### 6.41.4.20 __STATIC_INLINE uint32_t Chip_SPI_GetFIFOStatus ( LPC_SPI_T ∗ *pSPI* )

Get the current status of SPI controller FIFO.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

SPI Status (Or-ed bit value of SPI_FIFOSTAT_∗)

**Note**

Mask the return value with a value of type SPI_FIFOSTAT_∗ to determine if that status is active.

Definition at line 508 of file spi_common_5411x.h.

#### 6.41.4.21 __STATIC_INLINE uint32_t Chip_SPI_GetFIFOTrigLevel ( LPC_SPI_T ∗ *pSPI* )

Get SPI FIFO trigger-level.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base of on-chip SPI peripheral |

**Returns**

Returns the complete raw trigger register.

Definition at line 548 of file spi_common_5411x.h.

#### 6.41.4.22 __STATIC_INLINE uint32_t Chip_SPI_GetPendingInts ( LPC_SPI_T ∗ *pSPI* )

Return pending SPI interrupts.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

An Or'ed value of SPI_INT_∗ values

Definition at line 456 of file spi_common_5411x.h.

#### 6.41.4.23 __STATIC_INLINE uint32_t Chip_SPI_GetStatus ( LPC_SPI_T ∗ *pSPI* )

Get the current status of SPI controller.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

SPI Status (Or-ed bit value of SPI_STAT_∗)

**Note**

Mask the return value with a value of type SPI_STAT_∗ to determine if that status is active.

Definition at line 400 of file spi_common_5411x.h.

#### 6.41.4.24 int Chip_SPI_Init ( LPC_SPI_T ∗ *pSPI* )

Initialize the SPI.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base SPI peripheral on the chip |

**Returns**

Nothing

Definition at line 50 of file spi_common_5411x.c.

#### 6.41.4.25 __STATIC_INLINE uint32_t Chip_SPI_ReadRawRXFifo ( LPC_SPI_T ∗ *pSPI* )

Read raw data from receive FIFO with status bits.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Current value in receive data FIFO plus status bits

Definition at line 607 of file spi_common_5411x.h.

#### 6.41.4.26 __STATIC_INLINE uint16_t Chip_SPI_ReadRXData ( LPC_SPI_T ∗ *pSPI* )

Read data from receive FIFO masking off status bits.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |

**Returns**

Current value in receive data FIFO

**Note**

The entire register is read, but only the low 16-bits are returned.

Definition at line 620 of file spi_common_5411x.h.

#### 6.41.4.27 __STATIC_INLINE void Chip_SPI_SetCFGRegBits ( LPC_SPI_T ∗ *pSPI,* uint32_t *bits* )

Set SPI CFG register values.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *bits* | : CFG register bits to set, amd OR'ed value of SPI_CFG_∗ definitions |

**Returns**

Nothing

**Note**

This function safely sets only the selected bits in the SPI CFG register. It can be used to enable multiple bits at once.

Definition at line 259 of file spi_common_5411x.h.

#### 6.41.4.28 __STATIC_INLINE void Chip_SPI_SetCSPolHigh ( LPC_SPI_T ∗ *pSPI,* uint8_t *csNum* )

Set polarity on the SPI chip select high.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *csNum* | : Chip select number, 0 - 3 |

**Returns**

Nothing

**Note**

SPI chip select polarity is active high.

Definition at line 359 of file spi_common_5411x.h.

#### 6.41.4.29 __STATIC_INLINE void Chip_SPI_SetCSPolLow ( LPC_SPI_T ∗ *pSPI*, uint8_t *csNum* )

Set polarity on the SPI chip select low.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *csNum* | : Chip select number, 0 - 3 |

**Returns**

Nothing

**Note**

SPI chip select polarity is active low.

Definition at line 371 of file spi_common_5411x.h.

#### 6.41.4.30 __STATIC_INLINE void Chip_SPI_SetFIFOCfg ( LPC_SPI_T ∗ *pSPI*, uint32_t *cfg* )

Set FIFO Configuration register.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *cfg* | : Configuration value mask (OR'ed SPI_FIFOCFG_∗ values like SPI_FIFOCFG_ENABLETX) |

**Returns**

Nothing

Definition at line 482 of file spi_common_5411x.h.

#### 6.41.4.31 __STATIC_INLINE void Chip_SPI_SetFIFOTrigLevel ( LPC_SPI_T ∗ *pSPI*, uint8_t *tx_lvl*, uint8_t *rx_lvl* )

Setup SPI FIFO trigger-level.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *tx_lvl* | : TX Trigger level [Valid values 0 to 7] |
| *rx_lvl* | : RX Trigger level [Valid values 0 to 7] |

**Returns**

Nothing

**Note**

When *tx_lvl* = 0; trigger will happen when TX FIFO is empty if *tx_lvl* = 7; trigger will happen when TX FIFO has at least one free space
When *rx_lvl* = 0; trigger will happen when RX FIFO has at least one data in it, if *rx_lvl* = 7; trigger will happen when RX FIFO is full and cannot receive anymore data.

Definition at line 538 of file spi_common_5411x.h.

#### 6.41.4.32 __STATIC_INLINE void Chip_SPI_SetSPIMode ( LPC_SPI_T ∗ *pSPI*, SPI_CLOCK_MODE_T *mode* )

Set SPI mode.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *mode* | : SPI mode to set the SPI interface to |

**Returns**

Nothing

Definition at line 346 of file spi_common_5411x.h.

#### 6.41.4.33 __STATIC_INLINE void Chip_SPI_SetTXCTRLData ( LPC_SPI_T ∗ *pSPI*, uint16_t *ctrl*, uint16_t *data* )

Write FIFOWR register: writes control options and data.

**Parameters**

| | |
|---|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *ctrl* | : Control bits to be set |
| *data* | : Data to be transfered |

**Returns**

Nothing

**Note**

This function safely sets only set bits in FIFOWR control register. It can be used to enable multiple options at once.

Definition at line 648 of file spi_common_5411x.h.

#### 6.41.4.34 __STATIC_INLINE void Chip_SPI_WriteFIFO ( LPC_SPI_T ∗ *pSPI*, uint32_t *data* )

Write FIFOWR register: writes 32-bit value to FIFO.

**Parameters**

| | |
|---:|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *data* | : Control and Data to be transfered |

**Returns**

Nothing

Definition at line 634 of file spi_common_5411x.h.

#### 6.41.4.35 __STATIC_INLINE void Chip_SPI_WriteTXData ( LPC_SPI_T ∗ *pSPI,* uint16_t *data* )

Write data to transmit FIFO.

**Parameters**

| | |
|---:|---|
| *pSPI* | : Base on-chip SPI peripheral address |
| *data* | : Data to write |

**Returns**

Nothing

Definition at line 660 of file spi_common_5411x.h.

# 6.42 CHIP: LPC5411X SPI master driver

## 6.42.1 Detailed Description

### Data Structures

- struct SPIM_DELAY_CONFIG_T

  *SPI Delay Configure Struct.*
- struct SPIM_XFER_T

  *SPI Master transfer data context.*

### Macros

- #define SPIM_XFER_OPT_FRAME_DLY (1 << 0) /∗ frame delay between frames ∗/
- #define SPIM_XFER_OPT_FRAME_ASSERT (1 << 1) /∗ assert/de-assert ssel for each frame ∗/
- #define SPIM_XFER_OPT_DMA (1 << 2) /∗ use DMA ∗/

### Enumerations

- enum SPIM_EVENT_T {
  SPIM_EVENT_WAIT, SPIM_EVENT_ERRORTX, SPIM_EVENT_ERRORRX, SPIM_EVENT_ERROR,
  SPIM_EVENT_DONE }

  *SPI master Xfer events.*
- enum SPIM_XFER_STATE_T {
  SPI_XFER_STATE_IDLE, SPI_XFER_STATE_BUSY, SPI_XFER_STATE_DONE, SPI_XFER_STATE_STALL,
  SPI_XFER_STATE_ERROR }

  *States of SPI Master Xfer.*

### Functions

- __STATIC_INLINE uint32_t Chip_SPIM_GetClockRate (LPC_SPI_T ∗pSPI)

  *Get SPI master bit rate.*
- uint32_t Chip_SPIM_SetClockRate (LPC_SPI_T ∗pSPI, uint32_t rate)

  *Set SPI master bit rate.*
- void Chip_SPIM_DelayConfig (LPC_SPI_T ∗pSPI, SPIM_DELAY_CONFIG_T ∗pConfig)

  *Config SPI Delay parameters.*
- __STATIC_INLINE void Chip_SPIM_ForceEndOfTransfer (LPC_SPI_T ∗pSPI)

  *Forces an end of transfer for the current master transfer.*
- __STATIC_INLINE void Chip_SPIM_EnableLoopBack (LPC_SPI_T ∗pSPI)

  *Enable loopback mode.*
- __STATIC_INLINE void Chip_SPIM_DisableLoopBack (LPC_SPI_T ∗pSPI)

  *Disable loopback mode.*
- void Chip_SPIM_XferHandler (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *SPI master transfer state change handler.*
- void Chip_SPIM_Xfer (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Start non-blocking SPI master transfer.*
- void Chip_SPIM_XferFIFO (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Start polled SPI master transfer.*
- void Chip_SPIM_XferBlocking (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Perform blocking SPI master transfer.*

## 6.42.2 Macro Definition Documentation

### 6.42.2.1 #define SPIM_XFER_OPT_DMA (1 << 2) /* use DMA */

Definition at line 147 of file spim_5411x.h.

### 6.42.2.2 #define SPIM_XFER_OPT_FRAME_ASSERT (1 << 1) /* assert/de-assert ssel for each frame */

Definition at line 146 of file spim_5411x.h.

### 6.42.2.3 #define SPIM_XFER_OPT_FRAME_DLY (1 << 0) /* frame delay between frames */

Definition at line 145 of file spim_5411x.h.

## 6.42.3 Enumeration Type Documentation

### 6.42.3.1 enum SPIM_EVENT_T

SPI master Xfer events.

**Enumerator**

***SPIM_EVENT_WAIT*** SPI Xfer is waiting to be complete

***SPIM_EVENT_ERRORTX*** SPI TX underflow error

***SPIM_EVENT_ERRORRX*** SPI RX overflow error

***SPIM_EVENT_ERROR*** SPI Xfer ended with an error!

***SPIM_EVENT_DONE*** SPI Xfer completed without errors

Definition at line 126 of file spim_5411x.h.

### 6.42.3.2 enum SPIM_XFER_STATE_T

States of SPI Master Xfer.

**Enumerator**

***SPI_XFER_STATE_IDLE*** Transfer is idle and not started

***SPI_XFER_STATE_BUSY*** Transfer has started and in progress

***SPI_XFER_STATE_DONE*** Transfer is complete without errors

***SPI_XFER_STATE_STALL*** SPI Bus stalled

***SPI_XFER_STATE_ERROR*** SPI transfer terminated with errors

Definition at line 137 of file spim_5411x.h.

## 6.42.4 Function Documentation

### 6.42.4.1 void Chip_SPIM_DelayConfig ( LPC_SPI_T * *pSPI,* SPIM_DELAY_CONFIG_T * *pConfig* )

Config SPI Delay parameters.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *pConfig* | : SPI Delay Configure Struct (See SPIM_DELAY_CONFIG_T) |

**Returns**

Nothing

Definition at line 152 of file spim_5411x.c.

#### 6.42.4.2 __STATIC_INLINE void Chip_SPIM_DisableLoopBack ( LPC_SPI_T ∗ *pSPI* )

Disable loopback mode.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |

**Returns**

Nothing

Definition at line 116 of file spim_5411x.h.

#### 6.42.4.3 __STATIC_INLINE void Chip_SPIM_EnableLoopBack ( LPC_SPI_T ∗ *pSPI* )

Enable loopback mode.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |

**Returns**

Nothing

**Note**

Serial input is taken from the serial output (MOSI or MISO) rather than the serial input pin.

Definition at line 106 of file spim_5411x.h.

#### 6.42.4.4 __STATIC_INLINE void Chip_SPIM_ForceEndOfTransfer ( LPC_SPI_T ∗ *pSPI* )

Forces an end of transfer for the current master transfer.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |

**Returns**

Nothing

**Note**

Use this function to perform an immediate end of trasnfer for the current master operation. If the master is currently transferring data started with the Chip_SPIM_Xfer function, this terminates the transfer after the current byte completes and completes the transfer.

Definition at line 94 of file spim_5411x.h.

#### 6.42.4.5 __STATIC_INLINE uint32_t Chip_SPIM_GetClockRate ( LPC_SPI_T ∗ *pSPI* )

Get SPI master bit rate.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |

**Returns**

The actual SPI clock bit rate

Definition at line 51 of file spim_5411x.h.

#### 6.42.4.6 uint32_t Chip_SPIM_SetClockRate ( LPC_SPI_T ∗ *pSPI,* uint32_t *rate* )

Set SPI master bit rate.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *rate* | : Desired clock bit rate for the SPI interface |

**Returns**

The actual SPI clock bit rate

**Note**

This function will set the SPI clock divider to get closest to the desired rate as possible.

Definition at line 125 of file spim_5411x.c.

#### 6.42.4.7 void Chip_SPIM_Xfer ( LPC_SPI_T ∗ *pSPI,* SPIM_XFER_T ∗ *xfer* )

Start non-blocking SPI master transfer.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *xfer* | : Pointer to a SPIM_XFER_T structure see notes below |

**Returns**

Nothing

**Note**

This function starts a non-blocking SPI master transfer with the parameters setup in the passed SPIM_XFER_T structure. Once the transfer is started, the interrupt handler must call Chip_SPIM_XferHandler to keep the transfer going and fed with data. This function should only be called when the master is idle.

This function must be called with the options and sselNum fields correctly setup. Initial data buffers and the callback pointer must also be setup. No sanity checks are performed on the passed data.

Example call:
SPIM_XFER_T mxfer; mxfer.pCB = (&)masterCallbacks; mxfer.sselNum = 2; // Use chip select 2 mxfer.options =

SPI_TXDAT_FLEN(8); // 8 data bits, supports 1 - 16 bits mxfer.options |= SPI_TXDAT_EOT | SPI_TXDAT_EOF; // Apply frame and transfer delays to master transfer mxfer.options |= SPI_TXDAT_RXIGNORE; // Ignore RX data, will toss receive data regardless of pRXData8 or pRXData16 buffer mxfer.pTXData8 = SendBuffer; mxfer.txCount = 16; // Number of bytes to send before SPIMasterXferSend callback is called mxfer.pRXData8 = RecvBuffer; // Will not receive data if pRXData8/pRXData16 is NULL or SPI_TXDAT_RXIGNORE option is set mxfer.rxCount = 16; // Number of bytes to receive before SPIMasterXferRecv callback is called Chip_SPIM_Xfer(LPC_SPI0, &mxfer); // Start transfer

Note that the transfer, once started, needs to be constantly fed by the callbacks. The txCount and rxCount field only indicate the buffer size before the callbacks are called. To terminate the transfer, the SPIMasterXferSend callback must set the terminate field.

Definition at line 209 of file spim_5411x.c.

#### 6.42.4.8 void Chip_SPIM_XferBlocking ( **LPC_SPI_T** ∗ ***pSPI,*** **SPIM_XFER_T** ∗ ***xfer*** )

Perform blocking SPI master transfer.

**Parameters**

| | |
|---:|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *xfer* | : Pointer to a SPIM_XFER_T structure see notes below |

**Returns**

Nothing

**Note**

This function starts a blocking SPI master transfer with the parameters setup in the passed SPIM_XFER_T structure. Once the transfer is started, the callbacks in Chip_SPIM_XferHandler may be called to keep the transfer going and fed with data. SPI interrupts must be disabled prior to calling this function. It is not recommended to use this function.

Definition at line 275 of file spim_5411x.c.

#### 6.42.4.9 void Chip_SPIM_XferFIFO ( **LPC_SPI_T** ∗ ***pSPI,*** **SPIM_XFER_T** ∗ ***xfer*** )

Start polled SPI master transfer.

**Parameters**

| | |
|---:|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *xfer* | : Pointer to a SPIM_XFER_T structure see notes below |

**Returns**

Nothing

**Note**

This function starts a polled SPI master transfer with the parameters setup in the passed SPIM_XFER_T structure. This function should only be called when the master is idle.

This function forces the xfer counts to be smaller than the FIFO size. For small transfers, this is very efficient as there are no interrupts or DMA.

This function must be called with the options and sselNum fields set correctly Initial data buffers. The xfer transmit buffer is limited to the size of the FIFO.

Definition at line 240 of file spim_5411x.c.

#### 6.42.4.10 void Chip_SPIM_XferHandler ( LPC_SPI_T ∗ *pSPI,* SPIM_XFER_T ∗ *xfer* )

SPI master transfer state change handler.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *xfer* | : Pointer to a SPIM_XFER_T structure see notes below |

**Returns**

Nothing

**Note**

See SPIM_XFER_T for more information on this function. When using this function, the SPI master interrupts should be enabled and setup in the SPI interrupt handler to call this function when they fire. This function is meant to be called from the interrupt handler.

Definition at line 161 of file spim_5411x.c.

# 6.43 CHIP: LPC5411X SPI slave driver

## 6.43.1 Detailed Description

### Data Structures

- struct SPIS_XFER_T

### Enumerations

- enum SPIS_EVENT_T {
  SPIS_EVENT_SASSERT, SPIS_EVENT_SDEASSERT, SPIS_EVENT_DONE, SPIS_EVENT_ERRORTX, SPIS_EVENT_ERRORRX, SPIS_EVENT_THRESHOLD }

  *Slave callback events.*

### Functions

- void Chip_SPIS_Init (LPC_SPI_T ∗pSPI)

  *SPI slave initialization.*
- void Chip_SPIS_EnableInts (LPC_SPI_T ∗pSPI)

  *SPI slave interrupt enable.*
- void Chip_SPIS_DisableInts (LPC_SPI_T ∗pSPI)

  *SPI slave interrupt disable.*
- void Chip_SPIS_LoadFIFO (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *Load slave transmit FIFO.*
- void Chip_SPIS_ReadFIFO (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *SPI slave FIFO read.*
- void Chip_SPIS_XferHandler (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *SPI slave transfer state change handler.*

## 6.43.2 Enumeration Type Documentation

### 6.43.2.1 enum SPIS_EVENT_T

Slave callback events.

**Enumerator**

***SPIS_EVENT_SASSERT*** Slave select line asserted

***SPIS_EVENT_SDEASSERT*** Slave select line de-asserted

***SPIS_EVENT_DONE*** Slave done with this transfer [More data can be provided]

***SPIS_EVENT_ERRORTX*** TX Underflow error

***SPIS_EVENT_ERRORRX*** RX Overflow error

***SPIS_EVENT_THRESHOLD*** Slave transfer has reached its threshold

Definition at line 47 of file spis_5411x.h.

## 6.43.3 Function Documentation

### 6.43.3.1 void Chip_SPIS_DisableInts ( LPC_SPI_T ∗ *pSPI* )

SPI slave interrupt disable.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |

**Returns**

Nothing

**Note**

Clears / disables the following SPI interrupts: SPI_INT_SSAEN, SPI_INT_SSDEN. Clears / disables the following FIFO interrupts: SPI_FIFOINT_RXERR, SPI_FIFOINT_TXERR, SPI_FIFOINT_RXLVL, and SPI_FI↩ FOINT_TXLVL.

Definition at line 132 of file spis_5411x.c.

#### 6.43.3.2 void Chip_SPIS_EnableInts ( LPC_SPI_T ∗ *pSPI* )

SPI slave interrupt enable.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |

**Returns**

Nothing

**Note**

Clears / enables the following SPI interrupts: SPI_INT_SSAEN, SPI_INT_SSDEN. Clears / enables the following FIFO interrupts: SPI_FIFOINT_RXERR, SPI_FIFOINT_TXERR, SPI_FIFOINT_RXLVL, and SPI_FI↩ FOINT_TXLVL.

Definition at line 122 of file spis_5411x.c.

#### 6.43.3.3 void Chip_SPIS_Init ( LPC_SPI_T ∗ *pSPI* )

SPI slave initialization.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |

**Returns**

Nothing

**Note**

Initializes a slave SPI port.

Definition at line 110 of file spis_5411x.c.

#### 6.43.3.4 void Chip_SPIS_LoadFIFO ( LPC_SPI_T ∗ *pSPI,* SPIS_XFER_T ∗ *xfer* )

Load slave transmit FIFO.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *xfer* | : Pointer to a SPIS_XFER_T structure see notes below |

**Returns**

Nothing

**Note**

Prime the transmit FIFO with data prior to the master xfer start. If data is not pre-buffered, the initial slave transmit data will always be 0x0 with a slave transmit underflow status.

Definition at line 142 of file spis_5411x.c.

#### 6.43.3.5 void Chip_SPIS_ReadFIFO ( LPC_SPI_T ∗ *pSPI,* SPIS_XFER_T ∗ *xfer* )

SPI slave FIFO read.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *xfer* | : Pointer to a SPIS_XFER_T structure |

**Returns**

Nothing

**Note**

This function reads all RX FIFO data into the xfer RX buffer. Interrupts are not required and should be disabled prior to calling this function.

Definition at line 163 of file spis_5411x.c.

#### 6.43.3.6 void Chip_SPIS_XferHandler ( LPC_SPI_T ∗ *pSPI,* SPIS_XFER_T ∗ *xfer* )

SPI slave transfer state change handler.

**Parameters**

| | |
|---|---|
| *pSPI* | : The base of SPI peripheral on the chip |
| *xfer* | : Pointer to a SPIS_XFER_T structure see notes below |

**Returns**

Nothing

**Note**

See SPIS_XFER_T for more information on this function. When using this function, the SPI slave interrupts should be enabled and setup in the SPI interrupt handler to call this function when they fire. This function is meant to be called from the interrupt handler. The SPIS_XFER_T data does not need to be setup prior to the call and should be setup by the callbacks instead.

The callbacks are handled in the interrupt handler. If you are getting overflow or underflow errors, you might need to lower the speed of the master clock or extend the master's select assetion time.

Definition at line 177 of file spis_5411x.c.

## 6.44 CHIP: LPC5411X State Configurable Timer PWM driver

### 6.44.1 Detailed Description

For more information on how to use the driver please visit the FAQ page at www.lpcware.com

### Functions

- __STATIC_INLINE uint32_t Chip_SCTPWM_GetTicksPerCycle (LPC_SCT_T ∗pSCT)

  *Get number of ticks per PWM cycle.*
- __STATIC_INLINE uint32_t Chip_SCTPWM_PercentageToTicks (LPC_SCT_T ∗pSCT, uint8_t percent)

  *Converts a percentage to ticks.*
- __STATIC_INLINE uint32_t Chip_SCTPWM_GetDutyCycle (LPC_SCT_T ∗pSCT, uint8_t index)

  *Get number of ticks on per PWM cycle.*
- __STATIC_INLINE void Chip_SCTPWM_SetDutyCycle (LPC_SCT_T ∗pSCT, uint8_t index, uint32_t ticks)

  *Get number of ticks on per PWM cycle.*
- __STATIC_INLINE void Chip_SCTPWM_Init (LPC_SCT_T ∗pSCT)

  *Initialize the SCT/PWM clock and reset.*
- __STATIC_INLINE void Chip_SCTPWM_Start (LPC_SCT_T ∗pSCT)

  *Start the SCT PWM.*
- __STATIC_INLINE void Chip_SCTPWM_Stop (LPC_SCT_T ∗pSCT)

  *Stop the SCT PWM.*
- void Chip_SCTPWM_SetRate (LPC_SCT_T ∗pSCT, uint32_t freq)

  *Sets the frequency of the generated PWM wave.*
- void Chip_SCTPWM_SetOutPin (LPC_SCT_T ∗pSCT, uint8_t index, uint8_t pin)

  *Setup the OUTPUT pin and associate it with an index.*

### 6.44.2 Function Documentation

#### 6.44.2.1 __STATIC_INLINE uint32_t Chip_SCTPWM_GetDutyCycle ( LPC_SCT_T ∗ *pSCT,* uint8_t *index* )

Get number of ticks on per PWM cycle.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *index* | : Index of the PWM 1 to N (see notes) |

**Returns**

Number ot ticks for which the output will be ON per cycle

**Note**

*index* will be 1 to N where N is the "Number of match registers available in the SCT - 1" or "Number of OUTPUT pins available in the SCT" whichever is minimum.

Definition at line 86 of file sct_pwm_5411x.h.

#### 6.44.2.2 __STATIC_INLINE uint32_t Chip_SCTPWM_GetTicksPerCycle ( LPC_SCT_T ∗ *pSCT* )

Get number of ticks per PWM cycle.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |

**Returns**

Number ot ticks that will be counted per cycle

**Note**

Return value of this function will be vaild only after calling Chip_SCTPWM_SetRate()

Definition at line 56 of file sct_pwm_5411x.h.

#### 6.44.2.3 __STATIC_INLINE void Chip_SCTPWM_Init ( LPC_SCT_T ∗ *pSCT* )

Initialize the SCT/PWM clock and reset.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |

**Returns**

None

Definition at line 113 of file sct_pwm_5411x.h.

#### 6.44.2.4 __STATIC_INLINE uint32_t Chip_SCTPWM_PercentageToTicks ( LPC_SCT_T ∗ *pSCT,* uint8_t *percent* )

Converts a percentage to ticks.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *percent* | : Percentage to convert (0 - 100) |

**Returns**

Number ot ticks corresponding to given percentage

**Note**

Do not use this function when using very low pwm rate (like 100Hz or less), on a chip that has very high frequency as the calculation might cause integer overflow

Definition at line 71 of file sct_pwm_5411x.h.

#### 6.44.2.5 __STATIC_INLINE void Chip_SCTPWM_SetDutyCycle ( LPC_SCT_T ∗ *pSCT,* uint8_t *index,* uint32_t *ticks* )

Get number of ticks on per PWM cycle.

**Parameters**

| | |
|---:|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *index* | : Index of the PWM 1 to N (see notes) |
| *ticks* | : Number of ticks the output should say ON |

**Returns**

None

**Note**

*index* will be 1 to N where N is the "Number of match registers available in the SCT - 1" or "Number of OUTP←UT pins available in the SCT" whichever is minimum. The new duty cycle will be effective only after completion of current PWM cycle.

Definition at line 103 of file sct_pwm_5411x.h.

#### 6.44.2.6 void Chip_SCTPWM_SetOutPin ( LPC_SCT_T ∗ *pSCT,* uint8_t *index,* uint8_t *pin* )

Setup the OUTPUT pin and associate it with an index.

**Parameters**

| | |
|---:|---|
| *pSCT* | : The base of the SCT peripheral on the chip |
| *index* | : Index of PWM 1 to N (see notes) |
| *pin* | : COUT pin to be associated with the index |

**Returns**

None

**Note**

*index* will be 1 to N where N is the "Number of match registers available in the SCT - 1" or "Number of OUTPUT pins available in the SCT" whichever is minimum.

Definition at line 51 of file sct_pwm_5411x.c.

#### 6.44.2.7 void Chip_SCTPWM_SetRate ( LPC_SCT_T ∗ *pSCT,* uint32_t *freq* )

Sets the frequency of the generated PWM wave.

**Parameters**

| | |
|---:|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *freq* | : Frequency in Hz |

**Returns**

None

Definition at line 67 of file sct_pwm_5411x.c.

#### 6.44.2.8 __STATIC_INLINE void Chip_SCTPWM_Start ( LPC_SCT_T ∗ *pSCT* )

Start the SCT PWM.

**Parameters**

|  |  |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |

**Returns**

None

**Note**

This function must be called after all the configuration is completed. Do not call Chip_SCTPWM_SetRate() or Chip_SCTPWM_SetOutPin() after the SCT/PWM is started. Use Chip_SCTPWM_Stop() to stop the SCT/↩ PWM before reconfiguring, Chip_SCTPWM_SetDutyCycle() can be called when the SCT/PWM is running to change the DutyCycle.

Definition at line 129 of file sct_pwm_5411x.h.

#### 6.44.2.9 __STATIC_INLINE void Chip_SCTPWM_Stop ( LPC_SCT_T ∗ *pSCT* )

Stop the SCT PWM.

**Parameters**

|  |  |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |

**Returns**

None

Definition at line 139 of file sct_pwm_5411x.h.

## 6.45 CHIP: LPC5411X State Configurable Timer driver

### 6.45.1 Detailed Description

### Data Structures

- struct LPC_SCT_T

  *State Configurable Timer register block structure.*

### Macros

- #define CONFIG_SCT_nEV (13)
- #define CONFIG_SCT_nRG (13)
- #define CONFIG_SCT_nOU (8)
- #define CONFIG_SCT_nIN (8)
- #define SCT_CONFIG_16BIT_COUNTER 0x00000000

  *Macro defines for SCT configuration register.*
- #define SCT_CONFIG_32BIT_COUNTER 0x00000001
- #define SCT_CONFIG_CLKMODE_BUSCLK (0x0 << 1)
- #define SCT_CONFIG_CLKMODE_SCTCLK (0x1 << 1)
- #define SCT_CONFIG_CLKMODE_INCLK (0x2 << 1)
- #define SCT_CONFIG_CLKMODE_INEDGECLK (0x3 << 1)
- #define SCT_CONFIG_CLKMODE_SYSCLK (0x0 << 1)
- #define SCT_CONFIG_CLKMODE_PRESCALED_SYSCLK (0x1 << 1)
- #define SCT_CONFIG_CLKMODE_SCT_INPUT (0x2 << 1)
- #define SCT_CONFIG_CLKMODE_PRESCALED_SCT_INPUT (0x3 << 1)
- #define SCT_CONFIG_CKSEL_RISING_IN_0 (0x0UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_0 (0x1UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_1 (0x2UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_1 (0x3UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_2 (0x4UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_2 (0x5UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_3 (0x6UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_3 (0x7UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_4 (0x8UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_4 (0x9UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_5 (0xAUL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_5 (0xBUL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_6 (0xCUL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_6 (0xDUL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_7 (0xEUL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_7 (0xFUL << 3)
- #define SCT_CONFIG_NORELOADL_U (0x1 << 7)
- #define SCT_CONFIG_NORELOADH (0x1 << 8)
- #define SCT_CONFIG_AUTOLIMIT_U (0x1UL << 17)
- #define SCT_CONFIG_AUTOLIMIT_L (0x1UL << 17)
- #define SCT_CONFIG_AUTOLIMIT_H (0x1UL << 18)
- #define COUNTUP_TO_LIMIT_THEN_CLEAR_TO_ZERO 0

  *Macro defines for SCT control register.*
- #define COUNTUP_TO_LIMIT_THEN_CLEAR_TO_ZERO 0

  *Macro defines for SCT control register.*
- #define COUNTUP_TO LIMIT_THEN_COUNTDOWN_TO_ZERO 1
- #define COUNTUP_TO LIMIT_THEN_COUNTDOWN_TO_ZERO 1

- #define SCT_CTRL_STOP_L (1 << 1)
- #define SCT_CTRL_HALT_L (1 << 2)
- #define SCT_CTRL_CLRCTR_L (1 << 3)
- #define SCT_CTRL_BIDIR_L(x) (((x) & 0x01) << 4)
- #define SCT_CTRL_PRE_L(x) (((x) & 0xFF) << 5)
- #define SCT_CTRL_STOP_H (1 << 17)
- #define SCT_CTRL_HALT_H (1 << 18)
- #define SCT_CTRL_CLRCTR_H (1 << 19)
- #define SCT_CTRL_BIDIR_H(x) (((x) & 0x01) << 20)
- #define SCT_CTRL_PRE_H(x) (((x) & 0xFF) << 21)
- #define SCT_EV_CTRL_MATCHSEL(reg) (reg << 0)
- #define SCT_EV_CTRL_HEVENT_L (0UL << 4)
- #define SCT_EV_CTRL_HEVENT_H (1UL << 4)
- #define SCT_EV_CTRL_OUTSEL_INPUT (0UL << 5)
- #define SCT_EV_CTRL_OUTSEL_OUTPUT (0UL << 5)
- #define SCT_EV_CTRL_IOSEL(signal) (signal << 6)
- #define SCT_EV_CTRL_IOCOND_LOW (0UL << 10)
- #define SCT_EV_CTRL_IOCOND_RISE (0x1UL << 10)
- #define SCT_EV_CTRL_IOCOND_FALL (0x2UL << 10)
- #define SCT_EV_CTRL_IOCOND_HIGH (0x3UL << 10)
- #define SCT_EV_CTRL_COMBMODE_OR (0x0UL << 12)
- #define SCT_EV_CTRL_COMBMODE_MATCH (0x1UL << 12)
- #define SCT_EV_CTRL_COMBMODE_IO (0x2UL << 12)
- #define SCT_EV_CTRL_COMBMODE_AND (0x3UL << 12)
- #define SCT_EV_CTRL_STATELD (0x1UL << 14)
- #define SCT_EV_CTRL_STATEV(x) (x << 15)
- #define SCT_EV_CTRL_MATCHMEM (0x1UL << 20)
- #define SCT_EV_CTRL_DIRECTION_INDEPENDENT (0x0UL << 21)
- #define SCT_EV_CTRL_DIRECTION_UP (0x1UL << 21)
- #define SCT_EV_CTRL_DIRECTION_DOWN (0x2UL << 21)
- #define SCT_RES_NOCHANGE (0)

  *Macro defines for SCT Conflict resolution register.*
- #define SCT_RES_SET_OUTPUT (1)
- #define SCT_RES_CLEAR_OUTPUT (2)
- #define SCT_RES_TOGGLE_OUTPUT (3)

## Enumerations

- enum CHIP_SCT_MATCH_REG_T {
  SCT_MATCH_0 = 0, SCT_MATCH_1, SCT_MATCH_2, SCT_MATCH_3,
  SCT_MATCH_4, SCT_MATCH_5, SCT_MATCH_6, SCT_MATCH_7,
  SCT_MATCH_8, SCT_MATCH_9, SCT_MATCH_10, SCT_MATCH_11,
  SCT_MATCH_12, SCT_MATCH_13, SCT_MATCH_14, SCT_MATCH_15 }
- enum CHIP_SCT_EVENT_T {
  SCT_EVT_0 = (1 << 0), SCT_EVT_1 = (1 << 1), SCT_EVT_2 = (1 << 2), SCT_EVT_3 = (1 << 3),
  SCT_EVT_4 = (1 << 4), SCT_EVT_5 = (1 << 5), SCT_EVT_6 = (1 << 6), SCT_EVT_7 = (1 << 7),
  SCT_EVT_8 = (1 << 8), SCT_EVT_9 = (1 << 9), SCT_EVT_10 = (1 << 10), SCT_EVT_11 = (1 << 11),
  SCT_EVT_12 = (1 << 12), SCT_EVT_13 = (1 << 13), SCT_EVT_14 = (1 << 14), SCT_EVT_15 = (1 << 15) }

## Functions

- __STATIC_INLINE void Chip_SCT_EventControl (LPC_SCT_T ∗pSCT, uint32_t event_number, uint32_↵t value)

    *Set event control register.*
- __STATIC_INLINE void Chip_SCT_EventStateMask (LPC_SCT_T ∗pSCT, uint32_t event_number, uint32↵_t event_state_mask)

    *Set event state mask register.*
- __STATIC_INLINE void Chip_SCT_Config (LPC_SCT_T ∗pSCT, uint32_t cfg)

    *Set configuration register.*
- __STATIC_INLINE void Chip_SCT_Limit (LPC_SCT_T ∗pSCT, uint32_t value)

    *Configures the Limit register.*
- void Chip_SCT_SetClrControl (LPC_SCT_T ∗pSCT, uint32_t value, FunctionalState ena)

    *Set or Clear the Control register.*
- void Chip_SCT_SetConflictResolution (LPC_SCT_T ∗pSCT, uint8_t outnum, uint8_t value)

    *Set the conflict resolution.*
- __STATIC_INLINE void Chip_SCT_SetCount (LPC_SCT_T ∗pSCT, uint32_t count)

    *Set unified count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetCountL (LPC_SCT_T ∗pSCT, uint16_t count)

    *Set lower count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetCountH (LPC_SCT_T ∗pSCT, uint16_t count)

    *Set higher count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetMatchCount (LPC_SCT_T ∗pSCT, CHIP_SCT_MATCH_REG_T n, uint32_t value)

    *Set unified match count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetMatchReload (LPC_SCT_T ∗pSCT, CHIP_SCT_MATCH_REG_T n, uint32_t value)

    *Set unified match reload count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_EnableEventInt (LPC_SCT_T ∗pSCT, CHIP_SCT_EVENT_T evt)

    *Enable the interrupt for the specified event in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_DisableEventInt (LPC_SCT_T ∗pSCT, CHIP_SCT_EVENT_T evt)

    *Disable the interrupt for the specified event in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_ClearEventFlag (LPC_SCT_T ∗pSCT, CHIP_SCT_EVENT_T evt)

    *Clear the specified event flag in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetControl (LPC_SCT_T ∗pSCT, uint32_t value)

    *Set control register in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_ClearControl (LPC_SCT_T ∗pSCT, uint32_t value)

    *Clear control register in State Configurable Timer.*
- void Chip_SCT_Init (LPC_SCT_T ∗pSCT)

    *Initializes the State Configurable Timer.*
- void Chip_SCT_DeInit (LPC_SCT_T ∗pSCT)

    *Deinitializes the State Configurable Timer.*

## 6.45.2 Macro Definition Documentation

### 6.45.2.1 #define CONFIG_SCT_nEV (13)

Number of events

Definition at line 48 of file sct_5411x.h.

#### 6.45.2.2 #define CONFIG_SCT_nIN (8)

Number of outputs

Definition at line 51 of file sct_5411x.h.

#### 6.45.2.3 #define CONFIG_SCT_nOU (8)

Number of outputs

Definition at line 50 of file sct_5411x.h.

#### 6.45.2.4 #define CONFIG_SCT_nRG (13)

Number of match/compare registers

Definition at line 49 of file sct_5411x.h.

#### 6.45.2.5 #define COUNTUP_TO LIMIT_THEN_COUNTDOWN_TO_ZERO 1

Definition at line 224 of file sct_5411x.h.

#### 6.45.2.6 #define COUNTUP_TO LIMIT_THEN_COUNTDOWN_TO_ZERO 1

Definition at line 224 of file sct_5411x.h.

#### 6.45.2.7 #define COUNTUP_TO_LIMIT_THEN_CLEAR_TO_ZERO 0

Macro defines for SCT control register.

Direction for low or unified counter

Direction for high counter

Definition at line 223 of file sct_5411x.h.

#### 6.45.2.8 #define COUNTUP_TO_LIMIT_THEN_CLEAR_TO_ZERO 0

Macro defines for SCT control register.

Direction for low or unified counter

Direction for high counter

Definition at line 223 of file sct_5411x.h.

#### 6.45.2.9 #define SCT_CONFIG_16BIT_COUNTER 0x00000000

Macro defines for SCT configuration register.

Operate as 2 16-bit counters

Definition at line 176 of file sct_5411x.h.

#### 6.45.2.10 #define SCT_CONFIG_32BIT_COUNTER 0x00000001

Operate as 1 32-bit counter

Definition at line 177 of file sct_5411x.h.

#### 6.45.2.11 #define SCT_CONFIG_AUTOLIMIT_H (0x1UL $<<$ 18)

Definition at line 209 of file sct_5411x.h.

#### 6.45.2.12 #define SCT_CONFIG_AUTOLIMIT_L (0x1UL $<<$ 17)

Definition at line 208 of file sct_5411x.h.

#### 6.45.2.13 #define SCT_CONFIG_AUTOLIMIT_U (0x1UL $<<$ 17)

Definition at line 207 of file sct_5411x.h.

#### 6.45.2.14 #define SCT_CONFIG_CKSEL_FALLING_IN_0 (0x1UL $<<$ 3)

Definition at line 190 of file sct_5411x.h.

#### 6.45.2.15 #define SCT_CONFIG_CKSEL_FALLING_IN_1 (0x3UL $<<$ 3)

Definition at line 192 of file sct_5411x.h.

#### 6.45.2.16 #define SCT_CONFIG_CKSEL_FALLING_IN_2 (0x5UL $<<$ 3)

Definition at line 194 of file sct_5411x.h.

#### 6.45.2.17 #define SCT_CONFIG_CKSEL_FALLING_IN_3 (0x7UL $<<$ 3)

Definition at line 196 of file sct_5411x.h.

#### 6.45.2.18 #define SCT_CONFIG_CKSEL_FALLING_IN_4 (0x9UL $<<$ 3)

Definition at line 198 of file sct_5411x.h.

#### 6.45.2.19 #define SCT_CONFIG_CKSEL_FALLING_IN_5 (0xBUL $<<$ 3)

Definition at line 200 of file sct_5411x.h.

#### 6.45.2.20 #define SCT_CONFIG_CKSEL_FALLING_IN_6 (0xDUL $<<$ 3)

Definition at line 202 of file sct_5411x.h.

#### 6.45.2.21 #define SCT_CONFIG_CKSEL_FALLING_IN_7 (0xFUL $<<$ 3)

Definition at line 204 of file sct_5411x.h.

#### 6.45.2.22 #define SCT_CONFIG_CKSEL_RISING_IN_0 (0x0UL $<<$ 3)

Definition at line 189 of file sct_5411x.h.

#### 6.45.2.23 #define SCT_CONFIG_CKSEL_RISING_IN_1 (0x2UL << 3)

Definition at line 191 of file sct_5411x.h.

#### 6.45.2.24 #define SCT_CONFIG_CKSEL_RISING_IN_2 (0x4UL << 3)

Definition at line 193 of file sct_5411x.h.

#### 6.45.2.25 #define SCT_CONFIG_CKSEL_RISING_IN_3 (0x6UL << 3)

Definition at line 195 of file sct_5411x.h.

#### 6.45.2.26 #define SCT_CONFIG_CKSEL_RISING_IN_4 (0x8UL << 3)

Definition at line 197 of file sct_5411x.h.

#### 6.45.2.27 #define SCT_CONFIG_CKSEL_RISING_IN_5 (0xAUL << 3)

Definition at line 199 of file sct_5411x.h.

#### 6.45.2.28 #define SCT_CONFIG_CKSEL_RISING_IN_6 (0xCUL << 3)

Definition at line 201 of file sct_5411x.h.

#### 6.45.2.29 #define SCT_CONFIG_CKSEL_RISING_IN_7 (0xEUL << 3)

Definition at line 203 of file sct_5411x.h.

#### 6.45.2.30 #define SCT_CONFIG_CLKMODE_BUSCLK (0x0 << 1)

Bus clock

Definition at line 179 of file sct_5411x.h.

#### 6.45.2.31 #define SCT_CONFIG_CLKMODE_INCLK (0x2 << 1)

Input clock selected in CLKSEL field

Definition at line 181 of file sct_5411x.h.

#### 6.45.2.32 #define SCT_CONFIG_CLKMODE_INEDGECLK (0x3 << 1)

Input clock edge selected in CLKSEL field

Definition at line 182 of file sct_5411x.h.

#### 6.45.2.33 #define SCT_CONFIG_CLKMODE_PRESCALED_SCT_INPUT (0x3 << 1)

Prescaled input clock/edge selected in CKSEL field

Definition at line 187 of file sct_5411x.h.

#### 6.45.2.34 #define SCT_CONFIG_CLKMODE_PRESCALED_SYSCLK (0x1 << 1)

Prescaled system clock

Definition at line 185 of file sct_5411x.h.

#### 6.45.2.35 #define SCT_CONFIG_CLKMODE_SCT_INPUT (0x2 << 1)

Input clock/edge selected in CKSEL field

Definition at line 186 of file sct_5411x.h.

#### 6.45.2.36 #define SCT_CONFIG_CLKMODE_SCTCLK (0x1 << 1)

SCT clock

Definition at line 180 of file sct_5411x.h.

#### 6.45.2.37 #define SCT_CONFIG_CLKMODE_SYSCLK (0x0 << 1)

System clock

Definition at line 184 of file sct_5411x.h.

#### 6.45.2.38 #define SCT_CONFIG_NORELOADH (0x1 << 8)

Operate as 1 32-bit counter

Definition at line 206 of file sct_5411x.h.

#### 6.45.2.39 #define SCT_CONFIG_NORELOADL_U (0x1 << 7)

Operate as 1 32-bit counter

Definition at line 205 of file sct_5411x.h.

#### 6.45.2.40 #define SCT_CTRL_BIDIR_H( *x* ) (((x) & 0x01) << 20)

Definition at line 228 of file sct_5411x.h.

#### 6.45.2.41 #define SCT_CTRL_BIDIR_L( *x* ) (((x) & 0x01) << 4)

Bidirectional bit

Definition at line 220 of file sct_5411x.h.

#### 6.45.2.42 #define SCT_CTRL_CLRCTR_H (1 << 19)

Clear high counter

Definition at line 227 of file sct_5411x.h.

#### 6.45.2.43 #define SCT_CTRL_CLRCTR_L (1 $<<$ 3)

Clear low or unified counter

Definition at line 219 of file sct_5411x.h.

#### 6.45.2.44 #define SCT_CTRL_HALT_H (1 $<<$ 18)

Halt high counter

Definition at line 226 of file sct_5411x.h.

#### 6.45.2.45 #define SCT_CTRL_HALT_L (1 $<<$ 2)

Halt low counter

Definition at line 218 of file sct_5411x.h.

#### 6.45.2.46 #define SCT_CTRL_PRE_H( *x* ) (((x) & 0xFF) $<<$ 21)

Prescale clock for high counter

Definition at line 229 of file sct_5411x.h.

#### 6.45.2.47 #define SCT_CTRL_PRE_L( *x* ) (((x) & 0xFF) $<<$ 5)

Prescale clock for low or unified counter

Definition at line 221 of file sct_5411x.h.

#### 6.45.2.48 #define SCT_CTRL_STOP_H (1 $<<$ 17)

Stop high counter

Definition at line 225 of file sct_5411x.h.

#### 6.45.2.49 #define SCT_CTRL_STOP_L (1 $<<$ 1)

Stop low counter

Definition at line 217 of file sct_5411x.h.

#### 6.45.2.50 #define SCT_EV_CTRL_COMBMODE_AND (0x3UL $<<$ 12)

Definition at line 245 of file sct_5411x.h.

#### 6.45.2.51 #define SCT_EV_CTRL_COMBMODE_IO (0x2UL $<<$ 12)

Definition at line 244 of file sct_5411x.h.

#### 6.45.2.52 #define SCT_EV_CTRL_COMBMODE_MATCH (0x1UL $<<$ 12)

Definition at line 243 of file sct_5411x.h.

#### 6.45.2.53 #define SCT_EV_CTRL_COMBMODE_OR (0x0UL $<<$ 12)

Definition at line 242 of file sct_5411x.h.

#### 6.45.2.54 #define SCT_EV_CTRL_DIRECTION_DOWN (0x2UL $<<$ 21)

Definition at line 251 of file sct_5411x.h.

#### 6.45.2.55 #define SCT_EV_CTRL_DIRECTION_INDEPENDENT (0x0UL $<<$ 21)

Definition at line 249 of file sct_5411x.h.

#### 6.45.2.56 #define SCT_EV_CTRL_DIRECTION_UP (0x1UL $<<$ 21)

Definition at line 250 of file sct_5411x.h.

#### 6.45.2.57 #define SCT_EV_CTRL_HEVENT_H (1UL $<<$ 4)

Definition at line 233 of file sct_5411x.h.

#### 6.45.2.58 #define SCT_EV_CTRL_HEVENT_L (0UL $<<$ 4)

Definition at line 232 of file sct_5411x.h.

#### 6.45.2.59 #define SCT_EV_CTRL_IOCOND_FALL (0x2UL $<<$ 10)

Definition at line 240 of file sct_5411x.h.

#### 6.45.2.60 #define SCT_EV_CTRL_IOCOND_HIGH (0x3UL $<<$ 10)

Definition at line 241 of file sct_5411x.h.

#### 6.45.2.61 #define SCT_EV_CTRL_IOCOND_LOW (0UL $<<$ 10)

Definition at line 238 of file sct_5411x.h.

#### 6.45.2.62 #define SCT_EV_CTRL_IOCOND_RISE (0x1UL $<<$ 10)

Definition at line 239 of file sct_5411x.h.

#### 6.45.2.63 #define SCT_EV_CTRL_IOSEL( *signal* ) (signal $<<$ 6)

Definition at line 236 of file sct_5411x.h.

#### 6.45.2.64 #define SCT_EV_CTRL_MATCHMEM (0x1UL $<<$ 20)

Definition at line 248 of file sct_5411x.h.

#### 6.45.2.65 #define SCT_EV_CTRL_MATCHSEL( *reg* ) (reg << 0)

Definition at line 231 of file sct_5411x.h.

#### 6.45.2.66 #define SCT_EV_CTRL_OUTSEL_INPUT (0UL << 5)

Definition at line 234 of file sct_5411x.h.

#### 6.45.2.67 #define SCT_EV_CTRL_OUTSEL_OUTPUT (0UL << 5)

Definition at line 235 of file sct_5411x.h.

#### 6.45.2.68 #define SCT_EV_CTRL_STATELD (0x1UL << 14)

Definition at line 246 of file sct_5411x.h.

#### 6.45.2.69 #define SCT_EV_CTRL_STATEV( *x* ) (x << 15)

Definition at line 247 of file sct_5411x.h.

#### 6.45.2.70 #define SCT_RES_CLEAR_OUTPUT (2)

Definition at line 258 of file sct_5411x.h.

#### 6.45.2.71 #define SCT_RES_NOCHANGE (0)

Macro defines for SCT Conflict resolution register.

Definition at line 256 of file sct_5411x.h.

#### 6.45.2.72 #define SCT_RES_SET_OUTPUT (1)

Definition at line 257 of file sct_5411x.h.

#### 6.45.2.73 #define SCT_RES_TOGGLE_OUTPUT (3)

Definition at line 259 of file sct_5411x.h.

### 6.45.3 Enumeration Type Documentation

#### 6.45.3.1 enum CHIP_SCT_EVENT_T

SCT Event values enum

**Enumerator**

| | |
|---|---|
| ***SCT_EVT_0*** | Event 0 |
| ***SCT_EVT_1*** | Event 1 |
| ***SCT_EVT_2*** | Event 2 |
| ***SCT_EVT_3*** | Event 3 |
| ***SCT_EVT_4*** | Event 4 |

***SCT_EVT_5*** Event 5

***SCT_EVT_6*** Event 6

***SCT_EVT_7*** Event 7

***SCT_EVT_8*** Event 8

***SCT_EVT_9*** Event 9

***SCT_EVT_10*** Event 10

***SCT_EVT_11*** Event 11

***SCT_EVT_12*** Event 12

***SCT_EVT_13*** Event 13

***SCT_EVT_14*** Event 14

***SCT_EVT_15*** Event 15

Definition at line 286 of file sct_5411x.h.

#### 6.45.3.2 enum CHIP_SCT_MATCH_REG_T

SCT Match register values enum

**Enumerator**

***SCT_MATCH_0*** SCT Match register 0

***SCT_MATCH_1***

***SCT_MATCH_2***

***SCT_MATCH_3***

***SCT_MATCH_4***

***SCT_MATCH_5***

***SCT_MATCH_6***

***SCT_MATCH_7***

***SCT_MATCH_8***

***SCT_MATCH_9***

***SCT_MATCH_10***

***SCT_MATCH_11***

***SCT_MATCH_12***

***SCT_MATCH_13***

***SCT_MATCH_14***

***SCT_MATCH_15***

Definition at line 264 of file sct_5411x.h.

### 6.45.4 Function Documentation

#### 6.45.4.1 __STATIC_INLINE void Chip_SCT_ClearControl ( LPC_SCT_T ∗ *pSCT,* uint32_t *value* )

Clear control register in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *value* | : Value (ORed value of SCT_CTRL_∗ bits) |

**Returns**

Nothing

Definition at line 477 of file sct_5411x.h.

#### 6.45.4.2 __STATIC_INLINE void Chip_SCT_ClearEventFlag ( LPC_SCT_T ∗ *pSCT*, CHIP_SCT_EVENT_T *evt* )

Clear the specified event flag in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *evt* | : Event value |

**Returns**

Nothing

Definition at line 457 of file sct_5411x.h.

#### 6.45.4.3 __STATIC_INLINE void Chip_SCT_Config ( LPC_SCT_T ∗ *pSCT*, uint32_t *cfg* )

Set configuration register.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *cfg* | : The 32-bit configuration setting |

**Returns**

Nothing

Definition at line 335 of file sct_5411x.h.

#### 6.45.4.4 void Chip_SCT_DeInit ( LPC_SCT_T ∗ *pSCT* )

Deinitializes the State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |

**Returns**

Nothing

Definition at line 58 of file sct_5411x.c.

#### 6.45.4.5 __STATIC_INLINE void Chip_SCT_DisableEventInt ( LPC_SCT_T ∗ *pSCT*, CHIP_SCT_EVENT_T *evt* )

Disable the interrupt for the specified event in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *evt* | : Event value |

**Returns**

Nothing

Definition at line 447 of file sct_5411x.h.

### 6.45.4.6 __STATIC_INLINE void Chip_SCT_EnableEventInt ( LPC_SCT_T ∗ *pSCT*, CHIP_SCT_EVENT_T *evt* )

Enable the interrupt for the specified event in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *evt* | : Event value |

**Returns**

Nothing

Definition at line 437 of file sct_5411x.h.

### 6.45.4.7 __STATIC_INLINE void Chip_SCT_EventControl ( LPC_SCT_T ∗ *pSCT*, uint32_t *event_number*, uint32_t *value* )

Set event control register.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *event_number* | |
| *value* | : The 32-bit event control setting |

**Returns**

Nothing

Definition at line 312 of file sct_5411x.h.

### 6.45.4.8 __STATIC_INLINE void Chip_SCT_EventStateMask ( LPC_SCT_T ∗ *pSCT*, uint32_t *event_number*, uint32_t *event_state_mask* )

Set event state mask register.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *event_number* | |
| *event_state_↩ mask* | : The 32-bit event state mask setting |

**Returns**

Nothing

Definition at line 324 of file sct_5411x.h.

#### 6.45.4.9 void Chip_SCT_Init ( LPC_SCT_T ∗ *pSCT* )

Initializes the State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |

**Returns**

Nothing

Definition at line 51 of file sct_5411x.c.

#### 6.45.4.10 __STATIC_INLINE void Chip_SCT_Limit ( LPC_SCT_T ∗ *pSCT,* uint32_t *value* )

Configures the Limit register.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *value* | : The 32-bit Limit register value |

**Returns**

Nothing

Definition at line 345 of file sct_5411x.h.

#### 6.45.4.11 void Chip_SCT_SetClrControl ( LPC_SCT_T ∗ *pSCT,* uint32_t *value,* FunctionalState *ena* )

Set or Clear the Control register.

**Parameters**

| | |
|---|---|
| *pSCT* | : Pointer to SCT register block |
| *value* | : SCT Control register value |
| *ena* | : ENABLE - To set the fields specified by value : DISABLE - To clear the field specified by value |

**Returns**

Nothing Set or clear the control register bits as specified by the *value* parameter. If *ena* is set to ENABLE, the mentioned register fields will be set. If *ena* is set to DISABLE, the mentioned register fields will be cleared

Definition at line 64 of file sct_5411x.c.

#### 6.45.4.12 void Chip_SCT_SetConflictResolution ( LPC_SCT_T ∗ *pSCT,* uint8_t *outnum,* uint8_t *value* )

Set the conflict resolution.

**Parameters**

| | |
|---|---|
| *pSCT* | : Pointer to SCT register block |
| *outnum* | : Output number |
| *value* | : Output value<br>• SCT_RES_NOCHANGE :No change |

- SCT_RES_SET_OUTPUT :Set output
- SCT_RES_CLEAR_OUTPUT :Clear output

- SCT_RES_TOGGLE_OUTPUT :Toggle output : SCT_RES_NOCHANGE : DISABLE - To clear the field specified by value

  **Returns**

  Nothing Set conflict resolution for the output *outnum*

Definition at line 75 of file sct_5411x.c.

#### 6.45.4.13 __STATIC_INLINE void Chip_SCT_SetControl ( LPC_SCT_T ∗ *pSCT*, uint32_t *value* )

Set control register in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *value* | : Value (ORed value of SCT_CTRL_∗ bits) |

**Returns**

Nothing

Definition at line 467 of file sct_5411x.h.

#### 6.45.4.14 __STATIC_INLINE void Chip_SCT_SetCount ( LPC_SCT_T ∗ *pSCT*, uint32_t *count* )

Set unified count value in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *count* | : The 32-bit count value |

**Returns**

Nothing

Definition at line 385 of file sct_5411x.h.

#### 6.45.4.15 __STATIC_INLINE void Chip_SCT_SetCountH ( LPC_SCT_T ∗ *pSCT*, uint16_t *count* )

Set higher count value in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *count* | : The 16-bit count value |

**Returns**

Nothing

Definition at line 405 of file sct_5411x.h.

#### 6.45.4.16 __STATIC_INLINE void Chip_SCT_SetCountL ( LPC_SCT_T ∗ *pSCT*, uint16_t *count* )

Set lower count value in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *count* | : The 16-bit count value |

**Returns**

Nothing

Definition at line 395 of file sct_5411x.h.

#### 6.45.4.17 __STATIC_INLINE void Chip_SCT_SetMatchCount ( LPC_SCT_T ∗ *pSCT,* CHIP_SCT_MATCH_REG_T *n,* uint32_t *value* )

Set unified match count value in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *n* | : Match register value |
| *value* | : The 32-bit match count value |

**Returns**

Nothing

Definition at line 416 of file sct_5411x.h.

#### 6.45.4.18 __STATIC_INLINE void Chip_SCT_SetMatchReload ( LPC_SCT_T ∗ *pSCT,* CHIP_SCT_MATCH_REG_T *n,* uint32_t *value* )

Set unified match reload count value in State Configurable Timer.

**Parameters**

| | |
|---|---|
| *pSCT* | : The base of SCT peripheral on the chip |
| *n* | : Match register value |
| *value* | : The 32-bit match count reload value |

**Returns**

Nothing

Definition at line 427 of file sct_5411x.h.

# 6.46 CHIP: LPC5411X System and Control Driver

## 6.46.1 Detailed Description

### Data Structures

- struct LPC_SYSCON_T

  *LPC5411X Main system configuration register block structure.*
- struct LPC_ASYNC_SYSCON_T

  *LPC5411X Asynchronous system configuration register block structure.*

### Macros

- #define SYSCON_FROCTRL_MASK ((1 << 15) | (0xF << 26))

  *FROCTRL register bits.*
- #define SYSCON_FROCTRL_WRTRIM (1UL << 31)
- #define SYSCON_FROCTRL_HSPDCLK (1UL << 30)
- #define SYSCON_FROCTRL_USBMODCHG (1UL << 25)
- #define SYSCON_FROCTRL_USBCLKADJ (1UL << 24)
- #define SYSCON_FROCTRL_SEL96MHZ (1UL << 14)
- #define SYSCON_NMISRC_M0_ENABLE ((uint32_t) 1 << 30)
- #define SYSCON_NMISRC_M4_ENABLE ((uint32_t) 1 << 31)
- #define SYSCON_RST_POR (1 << 0)
- #define SYSCON_RST_EXTRST (1 << 1)
- #define SYSCON_RST_WDT (1 << 2)
- #define SYSCON_RST_BOD (1 << 3)
- #define SYSCON_RST_SYSRST (1 << 4)
- #define SYSCON_PDRUNCFG_PD_FRO (1 << 4)
- #define SYSCON_PDRUNCFG_PD_FLASH (1 << 5)
- #define SYSCON_PDRUNCFG_PD_TS (1 << 6)
- #define SYSCON_PDRUNCFG_PD_BOD_RST (1 << 7)
- #define SYSCON_PDRUNCFG_PD_BOD_INTR (1 << 8)
- #define SYSCON_PDRUNCFG_PD_ADC0 (1 << 10)
- #define SYSCON_PDRUNCFG_PD_VDDFLASH (1 << 11)
- #define SYSCON_PDRUNCFG_LP_VDDFLASH (1 << 12)
- #define SYSCON_PDRUNCFG_PD_SRAM0 (1 << 13)
- #define SYSCON_PDRUNCFG_PD_SRAM1 (1 << 14)
- #define SYSCON_PDRUNCFG_PD_SRAM2 (1 << 15)
- #define SYSCON_PDRUNCFG_PD_SRAMX (1 << 16)
- #define SYSCON_PDRUNCFG_PD_ROM (1 << 17)
- #define SYSCON_PDRUNCFG_PD_VDDHV_ENA (1 << 18)
- #define SYSCON_PDRUNCFG_PD_VDDA_ENA (1 << 19)
- #define SYSCON_PDRUNCFG_PD_WDT_OSC (1 << 20)
- #define SYSCON_PDRUNCFG_PD_USB_PHY (1 << 21)
- #define SYSCON_PDRUNCFG_PD_SYS_PLL (1 << 22)
- #define SYSCON_PDRUNCFG_PD_VREFP (1 << 23)
- #define SYSCON_AUTOCGOR_RAM0X (1 << 1)
- #define SYSCON_AUTOCGOR_RAM1 (1 << 2)
- #define SYSCON_AUTOCGOR_RAM2 (1 << 3)
- #define SYSCON_AUTOCGOR_MASK (SYSCON_AUTOCGOR_RAM0X | SYSCON_AUTOCGOR_RAM1 | SYSCON_AUTOCGOR_RAM2)

## Enumerations

- enum CHIP_SYSCON_BOOT_MODE_REMAP_T { REMAP_BOOT_LOADER_MODE, REMAP_USER_R↵ AM_MODE, REMAP_USER_FLASH_MODE }
- enum CHIP_SYSCON_PERIPH_RESET_T {
  RESET_FLASH = 7, RESET_FMC, RESET_SPIFI = 10, RESET_MUX,
  RESET_IOCON = 13, RESET_GPIO0, RESET_GPIO1, RESET_PINT = 18,
  RESET_GINT, RESET_DMA, RESET_CRC, RESET_WWDT,
  RESET_ADC = 27, RESET_ADC0 = 27, RESET_MRT = 32, RESET_SCT0 = 32 + 2,
  RESET_SCT = 32 + 2, RESET_UTICK = 32 + 10, RESET_FLEXCOMM0, RESET_FLEXCOMM1,
  RESET_FLEXCOMM2, RESET_FLEXCOMM3, RESET_FLEXCOMM4, RESET_FLEXCOMM5,
  RESET_FLEXCOMM6, RESET_FLEXCOMM7, RESET_DMIC, RESET_TIMER2 = 32 + 22,
  RESET_USB, RESET_TIMER0, RESET_TIMER1, RESET_TIMER3 = 128 + 13,
  RESET_TIMER4 }
- enum SYSCON_FLASHTIM_T {
  SYSCON_FLASH_1CYCLE = 0, FLASHTIM_20MHZ_CPU = SYSCON_FLASH_1CYCLE, SYSCON_FLA↵ SH_2CYCLE, SYSCON_FLASH_3CYCLE,
  SYSCON_FLASH_4CYCLE, SYSCON_FLASH_5CYCLE, SYSCON_FLASH_6CYCLE, SYSCON_FLAS↵ H_7CYCLE,
  SYSCON_FLASH_8CYCLE }

  *FLASH Access time definitions.*
- enum CHIP_SYSCON_WAKEUP_T {
  SYSCON_STARTER_WWDT_BOD = 0, SYSCON_STARTER_DMA, SYSCON_STARTER_GINT0, SYS↵ CON_STARTER_GINT1,
  SYSCON_STARTER_PINT0, SYSCON_STARTER_PINT1, SYSCON_STARTER_PINT2, SYSCON_STA↵ RTER_PINT3,
  SYSCON_STARTER_UTICK, SYSCON_STARTER_MRT, SYSCON_STARTER_TIMER0, SYSCON_ST↵ ARTER_TIMER1,
  SYSCON_STARTER_SCT0, SYSCON_STARTER_TIMER3, SYSCON_STARTER_FLEXCOMM0, SYSC↵ ON_STARTER_FLEXCOMM1,
  SYSCON_STARTER_FLEXCOMM2, SYSCON_STARTER_FLEXCOMM3, SYSCON_STARTER_FLEXC↵ OMM4, SYSCON_STARTER_FLEXCOMM5,
  SYSCON_STARTER_FLEXCOMM6, SYSCON_STARTER_FLEXCOMM7, SYSCON_STARTER_ADC0_↵ SEQA, SYSCON_STARTER_ADC0_SEQB,
  SYSCON_STARTER_ADC0_THCMP, SYSCON_STARTER_DMIC, SYSCON_STARTER_HWVAD, SYS↵ CON_STARTER_USBNEEDCLK,
  SYSCON_STARTER_USB, SYSCON_STARTER_RTC, SYSCON_STARTER_RESERVED0, SYSCON_↵ STARTER_MAILBOX,
  SYSCON_STARTER_PINT4, SYSCON_STARTER_PINT5, SYSCON_STARTER_PINT6, SYSCON_STA↵ RTER_PINT7,
  SYSCON_STARTER_TIMER2, SYSCON_STARTER_TIMER4 }

## Functions

- __STATIC_INLINE void Chip_SYSCON_Map (CHIP_SYSCON_BOOT_MODE_REMAP_T remap)

  *Re-map interrupt vectors.*
- __STATIC_INLINE CHIP_SYSCON_BOOT_MODE_REMAP_T Chip_SYSCON_GetMemoryMap (void)

  *Get system remap setting.*
- __STATIC_INLINE void Chip_SYSCON_SetSYSTCKCAL (uint32_t sysCalVal)

  *Set System tick timer calibration value.*
- void Chip_SYSCON_SetNMISource (uint32_t intsrc)

  *Set source for non-maskable interrupt (NMI)*
- void Chip_SYSCON_EnableNMISource (void)

  *Enable interrupt used for NMI source.*
- void Chip_SYSCON_DisableNMISource (void)

*Disable interrupt used for NMI source.*
- void Chip_SYSCON_Enable_ASYNC_Syscon (bool enable)

  *Enable or disable asynchronous APB bridge and subsystem.*
- __STATIC_INLINE void Chip_SYSCON_SetUSARTFRGCtrl (uint8_t fmul, uint8_t fdiv)

  *Set UART Fractional divider value.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetSystemRSTStatus (void)

  *Get system reset status.*
- __STATIC_INLINE void Chip_SYSCON_ClearSystemRSTStatus (uint32_t reset)

  *Clear system reset status.*
- __STATIC_INLINE void Chip_SYSCON_PeriphReset (CHIP_SYSCON_PERIPH_RESET_T periph)

  *Resets a peripheral.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetPORPIOStatus (uint8_t port)

  *Read POR captured PIO status.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetResetPIOStatus (uint8_t port)

  *Read reset captured PIO status.*
- __STATIC_INLINE void Chip_SYSCON_StartFreqMeas (void)

  *Starts a frequency measurement cycle.*
- __STATIC_INLINE bool Chip_SYSCON_IsFreqMeasComplete (void)

  *Indicates when a frequency measurement cycle is complete.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetRawFreqMeasCapval (void)

  *Returns the raw capture value for a frequency measurement cycle.*
- uint32_t Chip_SYSCON_GetCompFreqMeas (uint32_t refClockRate)

  *Returns the computed value for a frequency measurement cycle.*
- __STATIC_INLINE void Chip_SYSCON_SetFLASHAccess (SYSCON_FLASHTIM_T clks)

  *Set FLASH memory access time in clocks.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetPowerStates (void)

  *Power up one or more blocks or peripherals.*
- __STATIC_INLINE void Chip_SYSCON_PowerDown (uint32_t powerdownmask)

  *Power down one or more blocks or peripherals.*
- void Chip_SYSCON_PowerUp (uint32_t powerupmask)

  *Power up one or more blocks or peripherals.*
- __STATIC_INLINE void Chip_SYSCON_EnableWakeup (CHIP_SYSCON_WAKEUP_T periphId)

  *Enables a pin's (PINT) wakeup logic.*
- __STATIC_INLINE void Chip_SYSCON_DisableWakeup (CHIP_SYSCON_WAKEUP_T periphId)

  *Disables peripheral's wakeup logic.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetDeviceID (void)

  *Return the pointer to device ID registers.*
- __STATIC_INLINE void Chip_SYSCON_DisableAutoClocking (uint32_t mask)

  *Disables Auto clock gating for SRAM's.*
- __STATIC_INLINE void Chip_SYSCON_EnableAutoClocking (uint32_t mask)

  *Re-enables Auto clock gating for SRAM's.*

## 6.46.2 Macro Definition Documentation

### 6.46.2.1 #define SYSCON_AUTOCGOR_MASK (SYSCON_AUTOCGOR_RAM0X | SYSCON_AUTOCGOR_RAM1 | SYSCON_AUTOCGOR_RAM2)

Definition at line 579 of file syscon_5411x.h.

### 6.46.2.2 #define SYSCON_AUTOCGOR_RAM0X (1 $<<$ 1)

Auto Clock Gating Overide definition bitsRAM0 and RAMX overide

Definition at line 576 of file syscon_5411x.h.

### 6.46.2.3 #define SYSCON_AUTOCGOR_RAM1 (1 $<<$ 2)

RAM1 overide

Definition at line 577 of file syscon_5411x.h.

### 6.46.2.4 #define SYSCON_AUTOCGOR_RAM2 (1 $<<$ 3)

RAM2 overide

Definition at line 578 of file syscon_5411x.h.

### 6.46.2.5 #define SYSCON_FROCTRL_HSPDCLK (1UL $<<$ 30)

High speed clock (FROHF) enable bit

Definition at line 170 of file syscon_5411x.h.

### 6.46.2.6 #define SYSCON_FROCTRL_MASK ((1 $<<$ 15) $|$ (0xF $<<$ 26))

FROCTRL register bits.

MASK for reserved bits in FROCTRL register

Definition at line 168 of file syscon_5411x.h.

### 6.46.2.7 #define SYSCON_FROCTRL_SEL96MHZ (1UL $<<$ 14)

When set FROHF will be 96MHz; else FROHF will be 48MHz

Definition at line 173 of file syscon_5411x.h.

### 6.46.2.8 #define SYSCON_FROCTRL_USBCLKADJ (1UL $<<$ 24)

Automatically adjust FRO trim value based on SOF from USB

Definition at line 172 of file syscon_5411x.h.

### 6.46.2.9 #define SYSCON_FROCTRL_USBMODCHG (1UL $<<$ 25)

When set Trim value is pending to be set by SOF from USB

Definition at line 171 of file syscon_5411x.h.

### 6.46.2.10 #define SYSCON_FROCTRL_WRTRIM (1UL $<<$ 31)

Enable Writes to FROCTRL register

Definition at line 169 of file syscon_5411x.h.

#### 6.46.2.11 #define SYSCON_NMISRC_M0_ENABLE ((uint32_t) 1 << 30)

Non-Maskable Interrupt Enable/Disable valueEnable the Non-Maskable Interrupt M0 (NMI) source

Definition at line 215 of file syscon_5411x.h.

#### 6.46.2.12 #define SYSCON_NMISRC_M4_ENABLE ((uint32_t) 1 << 31)

Enable the Non-Maskable Interrupt M4 (NMI) source

Definition at line 216 of file syscon_5411x.h.

#### 6.46.2.13 #define SYSCON_PDRUNCFG_LP_VDDFLASH (1 << 12)

Flash LP Vdd

Definition at line 458 of file syscon_5411x.h.

#### 6.46.2.14 #define SYSCON_PDRUNCFG_PD_ADC0 (1 << 10)

ADC0

Definition at line 456 of file syscon_5411x.h.

#### 6.46.2.15 #define SYSCON_PDRUNCFG_PD_BOD_INTR (1 << 8)

Brown-out Detect interrupt

Definition at line 455 of file syscon_5411x.h.

#### 6.46.2.16 #define SYSCON_PDRUNCFG_PD_BOD_RST (1 << 7)

Brown-out Detect reset

Definition at line 454 of file syscon_5411x.h.

#### 6.46.2.17 #define SYSCON_PDRUNCFG_PD_FLASH (1 << 5)

Flash memory

Definition at line 452 of file syscon_5411x.h.

#### 6.46.2.18 #define SYSCON_PDRUNCFG_PD_FRO (1 << 4)

Power control definition bits (0 = powered, 1 = powered down)FRO oscillator

Definition at line 451 of file syscon_5411x.h.

#### 6.46.2.19 #define SYSCON_PDRUNCFG_PD_ROM (1 << 17)

ROM

Definition at line 463 of file syscon_5411x.h.

#### 6.46.2.20 #define SYSCON_PDRUNCFG_PD_SRAM0 (1 << 13)

SRAM0

Definition at line 459 of file syscon_5411x.h.

#### 6.46.2.21 #define SYSCON_PDRUNCFG_PD_SRAM1 (1 << 14)

SRAM1

Definition at line 460 of file syscon_5411x.h.

#### 6.46.2.22 #define SYSCON_PDRUNCFG_PD_SRAM2 (1 << 15)

SRAM2

Definition at line 461 of file syscon_5411x.h.

#### 6.46.2.23 #define SYSCON_PDRUNCFG_PD_SRAMX (1 << 16)

SRAMX

Definition at line 462 of file syscon_5411x.h.

#### 6.46.2.24 #define SYSCON_PDRUNCFG_PD_SYS_PLL (1 << 22)

PLL0

Definition at line 468 of file syscon_5411x.h.

#### 6.46.2.25 #define SYSCON_PDRUNCFG_PD_TS (1 << 6)

Temperature Sensor

Definition at line 453 of file syscon_5411x.h.

#### 6.46.2.26 #define SYSCON_PDRUNCFG_PD_USB_PHY (1 << 21)

USB Phy

Definition at line 467 of file syscon_5411x.h.

#### 6.46.2.27 #define SYSCON_PDRUNCFG_PD_VDDA_ENA (1 << 19)

Vdda to the ADC, must be enabled for the ADC to work

Definition at line 465 of file syscon_5411x.h.

#### 6.46.2.28 #define SYSCON_PDRUNCFG_PD_VDDFLASH (1 << 11)

Flash Vdd

Definition at line 457 of file syscon_5411x.h.

#### 6.46.2.29 #define SYSCON_PDRUNCFG_PD_VDDHV_ENA (1 << 18)

Vdd HV

Definition at line 464 of file syscon_5411x.h.

#### 6.46.2.30 #define SYSCON_PDRUNCFG_PD_VREFP (1 << 23)

Vrefp to the ADC, must be enabled for the ADC to work

Definition at line 469 of file syscon_5411x.h.

#### 6.46.2.31 #define SYSCON_PDRUNCFG_PD_WDT_OSC (1 << 20)

Watchdog oscillator

Definition at line 466 of file syscon_5411x.h.

#### 6.46.2.32 #define SYSCON_RST_BOD (1 << 3)

Brown-out detect reset status

Definition at line 265 of file syscon_5411x.h.

#### 6.46.2.33 #define SYSCON_RST_EXTRST (1 << 1)

External reset status

Definition at line 263 of file syscon_5411x.h.

#### 6.46.2.34 #define SYSCON_RST_POR (1 << 0)

System reset status valuesPOR reset status

Definition at line 262 of file syscon_5411x.h.

#### 6.46.2.35 #define SYSCON_RST_SYSRST (1 << 4)

software system reset status

Definition at line 266 of file syscon_5411x.h.

#### 6.46.2.36 #define SYSCON_RST_WDT (1 << 2)

Watchdog reset status

Definition at line 264 of file syscon_5411x.h.

### 6.46.3 Enumeration Type Documentation

#### 6.46.3.1 enum CHIP_SYSCON_BOOT_MODE_REMAP_T

System memory remap modes used to remap interrupt vectors

**Enumerator**

***REMAP_BOOT_LOADER_MODE*** Interrupt vectors are re-mapped to Boot ROM

***REMAP_USER_RAM_MODE*** Interrupt vectors are re-mapped to user Static RAM

***REMAP_USER_FLASH_MODE*** Interrupt vectors are not re-mapped and reside in Flash

Definition at line 177 of file syscon_5411x.h.

#### 6.46.3.2 enum CHIP_SYSCON_PERIPH_RESET_T

Peripheral reset identifiers

**Enumerator**

***RESET_FLASH*** Flash Controller

***RESET_FMC*** Flash Accelerator

***RESET_SPIFI*** SPIFI Reset

***RESET_MUX*** IO MUX Reset

***RESET_IOCON*** IOCON Reset

***RESET_GPIO0*** GPIO Port-0 Reset

***RESET_GPIO1*** GPIO Port-1 Reset

***RESET_PINT*** Pin Interrupt Reset

***RESET_GINT*** Group Interrupt Reset

***RESET_DMA*** DMA Reset

***RESET_CRC*** CRC Engine Reset

***RESET_WWDT*** Windowed watchdog timer

***RESET_ADC*** ADC Reset

***RESET_ADC0*** ADC Reset

***RESET_MRT*** Multirate Timer

***RESET_SCT0*** State configurable Timer

***RESET_SCT*** State configurable Timer

***RESET_UTICK*** Micro Tick Timer

***RESET_FLEXCOMM0*** FlexComm 0

***RESET_FLEXCOMM1*** FlexComm 1

***RESET_FLEXCOMM2*** FlexComm 2

***RESET_FLEXCOMM3*** FlexComm 3

***RESET_FLEXCOMM4*** FlexComm 4

***RESET_FLEXCOMM5*** FlexComm 5

***RESET_FLEXCOMM6*** FlexComm 6

***RESET_FLEXCOMM7*** FlexComm 7

***RESET_DMIC*** Digital MIC

***RESET_TIMER2*** Timer2 Reset

***RESET_USB*** USB Reset

***RESET_TIMER0*** Timer0 Reset

***RESET_TIMER1*** Timer1 Reset

***RESET_TIMER3*** TIMER0

***RESET_TIMER4*** TIMER4 Reset

Definition at line 292 of file syscon_5411x.h.

#### 6.46.3.3 enum CHIP_SYSCON_WAKEUP_T

Start enable enumerations - for enabling and disabling peripheral wakeup

**Enumerator**

***SYSCON_STARTER_WWDT_BOD***

***SYSCON_STARTER_DMA***

***SYSCON_STARTER_GINT0***

***SYSCON_STARTER_GINT1***

***SYSCON_STARTER_PINT0***

***SYSCON_STARTER_PINT1***

***SYSCON_STARTER_PINT2***

***SYSCON_STARTER_PINT3***

***SYSCON_STARTER_UTICK***

***SYSCON_STARTER_MRT***

***SYSCON_STARTER_TIMER0***

***SYSCON_STARTER_TIMER1***

***SYSCON_STARTER_SCT0***

***SYSCON_STARTER_TIMER3***

***SYSCON_STARTER_FLEXCOMM0***

***SYSCON_STARTER_FLEXCOMM1***

***SYSCON_STARTER_FLEXCOMM2***

***SYSCON_STARTER_FLEXCOMM3***

***SYSCON_STARTER_FLEXCOMM4***

***SYSCON_STARTER_FLEXCOMM5***

***SYSCON_STARTER_FLEXCOMM6***

***SYSCON_STARTER_FLEXCOMM7***

***SYSCON_STARTER_ADC0_SEQA***

***SYSCON_STARTER_ADC0_SEQB***

***SYSCON_STARTER_ADC0_THCMP***

***SYSCON_STARTER_DMIC***

***SYSCON_STARTER_HWVAD***

***SYSCON_STARTER_USBNEEDCLK***

***SYSCON_STARTER_USB***

***SYSCON_STARTER_RTC***

***SYSCON_STARTER_RESERVED0***

***SYSCON_STARTER_MAILBOX***

***SYSCON_STARTER_PINT4***

***SYSCON_STARTER_PINT5***

***SYSCON_STARTER_PINT6***

***SYSCON_STARTER_PINT7***

***SYSCON_STARTER_TIMER2***

***SYSCON_STARTER_TIMER4***

Definition at line 502 of file syscon_5411x.h.

#### 6.46.3.4 enum SYSCON_FLASHTIM_T

FLASH Access time definitions.

**Enumerator**

| | |
|---|---|
| ***SYSCON_FLASH_1CYCLE*** | Flash accesses use 1 CPU clock |
| ***FLASHTIM_20MHZ_CPU*** | |
| ***SYSCON_FLASH_2CYCLE*** | Flash accesses use 2 CPU clocks |
| ***SYSCON_FLASH_3CYCLE*** | Flash accesses use 3 CPU clocks |
| ***SYSCON_FLASH_4CYCLE*** | Flash accesses use 4 CPU clocks |
| ***SYSCON_FLASH_5CYCLE*** | Flash accesses use 5 CPU clocks |
| ***SYSCON_FLASH_6CYCLE*** | Flash accesses use 6 CPU clocks |
| ***SYSCON_FLASH_7CYCLE*** | Flash accesses use 7 CPU clocks |
| ***SYSCON_FLASH_8CYCLE*** | Flash accesses use 8 CPU clocks |

Definition at line 421 of file syscon_5411x.h.

### 6.46.4 Function Documentation

#### 6.46.4.1 __STATIC_INLINE void Chip_SYSCON_ClearSystemRSTStatus ( uint32_t *reset* )

Clear system reset status.

**Parameters**

| | |
|---|---|
| *reset* | : An Or'ed value of SYSCON_RST_∗ status to clear |

**Returns**

Nothing

**Note**

This function clears the specified reset source(s).

Definition at line 284 of file syscon_5411x.h.

#### 6.46.4.2 __STATIC_INLINE void Chip_SYSCON_DisableAutoClocking ( uint32_t *mask* )

Disables Auto clock gating for SRAM's.

**Parameters**

| | |
|---|---|
| *mask* | : Mask of RAM bits |

**Returns**

Nothing

Definition at line 586 of file syscon_5411x.h.

#### 6.46.4.3 void Chip_SYSCON_DisableNMISource ( void )

Disable interrupt used for NMI source.

**Returns**

Nothing

Definition at line 79 of file syscon_5411x.c.

#### 6.46.4.4 __STATIC_INLINE void Chip_SYSCON_DisableWakeup ( CHIP_SYSCON_WAKEUP_T *periphId* )

Disables peripheral's wakeup logic.

**Parameters**

| | |
|---|---|
| *periphId* | : Peripheral identifier |

**Returns**

Nothing

Definition at line 558 of file syscon_5411x.h.

#### 6.46.4.5 void Chip_SYSCON_Enable_ASYNC_Syscon ( bool *enable* )

Enable or disable asynchronous APB bridge and subsystem.

**Parameters**

| | |
|---|---|
| *enable* | : true to enable, false to disable |

**Returns**

Nothing

**Note**

This bridge must be enabled to access peripherals on the associated bridge.

Definition at line 89 of file syscon_5411x.c.

#### 6.46.4.6 __STATIC_INLINE void Chip_SYSCON_EnableAutoClocking ( uint32_t *mask* )

Re-enables Auto clock gating for SRAM's.

**Parameters**

| | |
|---|---|
| *mask* | : Mask of RAM bits |

**Returns**

Nothing

Definition at line 596 of file syscon_5411x.h.

#### 6.46.4.7 void Chip_SYSCON_EnableNMISource ( void )

Enable interrupt used for NMI source.

**Returns**

Nothing

Definition at line 69 of file syscon_5411x.c.

#### 6.46.4.8 __STATIC_INLINE void Chip_SYSCON_EnableWakeup ( CHIP_SYSCON_WAKEUP_T *periphId* )

Enables a pin's (PINT) wakeup logic.

**Parameters**

| | |
|---|---|
| *periphId* | : Peripheral identifier (See CHIP_SYSCON_WAKEUP_T) |

**Returns**

Nothing

Definition at line 548 of file syscon_5411x.h.

#### 6.46.4.9 uint32_t Chip_SYSCON_GetCompFreqMeas ( uint32_t *refClockRate* )

Returns the computed value for a frequency measurement cycle.

**Parameters**

| | |
|---|---|
| *refClockRate* | : Reference clock rate used during the frequency measurement cycle |

**Returns**

Computed cpature value

Definition at line 100 of file syscon_5411x.c.

#### 6.46.4.10 __STATIC_INLINE uint32_t Chip_SYSCON_GetDeviceID ( void )

Return the pointer to device ID registers.

**Returns**

Pointer to device ID registers

Definition at line 567 of file syscon_5411x.h.

#### 6.46.4.11 __STATIC_INLINE CHIP_SYSCON_BOOT_MODE_REMAP_T Chip_SYSCON_GetMemoryMap ( void )

Get system remap setting.

**Returns**

System remap setting

Definition at line 197 of file syscon_5411x.h.

#### 6.46.4.12 __STATIC_INLINE uint32_t Chip_SYSCON_GetPORPIOStatus ( uint8_t *port* )

Read POR captured PIO status.

**Parameters**

| | |
|---|---|
| *port* | : 0 for port 0 pins, 1 for port 1 pins, 2 for port 2 pins, etc. |

**Returns**

captured Power-On-Reset (POR) PIO status

Definition at line 365 of file syscon_5411x.h.

#### 6.46.4.13 __STATIC_INLINE uint32_t Chip_SYSCON_GetPowerStates ( void )

Power up one or more blocks or peripherals.

**Returns**

OR'ed values of SYSCON_PDRUNCFG_* values

**Note**

A high state indicates the peripheral is powered down.

Definition at line 476 of file syscon_5411x.h.

#### 6.46.4.14 __STATIC_INLINE uint32_t Chip_SYSCON_GetRawFreqMeasCapval ( void )

Returns the raw capture value for a frequency measurement cycle.

**Returns**

raw cpature value (this is not a frequency)

Definition at line 406 of file syscon_5411x.h.

#### 6.46.4.15 __STATIC_INLINE uint32_t Chip_SYSCON_GetResetPIOStatus ( uint8_t *port* )

Read reset captured PIO status.

**Parameters**

| | |
|---|---|
| *port* | : 0 for port 0 pins, 1 for port 1 pins, 2 for port 2 pins, etc. |

**Returns**

captured reset PIO status

**Note**

Used when reset other than a Power-On-Reset (POR) occurs.

Definition at line 376 of file syscon_5411x.h.

#### 6.46.4.16 __STATIC_INLINE uint32_t Chip_SYSCON_GetSystemRSTStatus ( void )

Get system reset status.

**Returns**

An Or'ed value of SYSCON_RST_*

**Note**

This function returns the detected reset source(s).

Definition at line 273 of file syscon_5411x.h.

#### 6.46.4.17 __STATIC_INLINE bool Chip_SYSCON_IsFreqMeasComplete ( void )

Indicates when a frequency measurement cycle is complete.

**Returns**

true if a measurement cycle is active, otherwise false

Definition at line 397 of file syscon_5411x.h.

#### 6.46.4.18 __STATIC_INLINE void Chip_SYSCON_Map ( CHIP_SYSCON_BOOT_MODE_REMAP_T *remap* )

Re-map interrupt vectors.

**Parameters**

| | |
|---|---|
| *remap* | : system memory map value |

**Returns**

Nothing

Definition at line 188 of file syscon_5411x.h.

#### 6.46.4.19 __STATIC_INLINE void Chip_SYSCON_PeriphReset ( CHIP_SYSCON_PERIPH_RESET_T *periph* )

Resets a peripheral.

**Parameters**

| | |
|---|---|
| *periph* | : Peripheral to reset (See CHIP_SYSCON_PERIPH_RESET_T) |

**Returns**

Nothing Will assert and de-assert reset for a peripheral.

Definition at line 339 of file syscon_5411x.h.

#### 6.46.4.20 __STATIC_INLINE void Chip_SYSCON_PowerDown ( uint32_t *powerdownmask* )

Power down one or more blocks or peripherals.

**Parameters**

| | |
|---|---|
| *powerdownmask* | : OR'ed values of SYSCON_PDRUNCFG_∗ values |

**Returns**

Nothing

Definition at line 486 of file syscon_5411x.h.

#### 6.46.4.21 void Chip_SYSCON_PowerUp ( uint32_t *powerupmask* )

Power up one or more blocks or peripherals.

**Parameters**

| | |
|---|---|
| *powerupmask* | : OR'ed values of SYSCON_PDRUNCFG_∗ values |

**Returns**

Nothing

Definition at line 116 of file syscon_5411x.c.

#### 6.46.4.22 __STATIC_INLINE void Chip_SYSCON_SetFLASHAccess ( SYSCON_FLASHTIM_T *clks* )

Set FLASH memory access time in clocks.

**Parameters**

| | |
|---|---|
| *clks* | : Clock cycles for FLASH access |

**Returns**

Nothing

Definition at line 438 of file syscon_5411x.h.

#### 6.46.4.23 void Chip_SYSCON_SetNMISource ( uint32_t *intsrc* )

Set source for non-maskable interrupt (NMI)

**Parameters**

| | |
|---|---|
| *intsrc* | : IRQ number to assign to the NMI |

**Returns**

Nothing

**Note**

The NMI source will be disabled upon exiting this function. Use the Chip_SYSCON_EnableNMISource() function to enable the NMI source.

Definition at line 51 of file syscon_5411x.c.

#### 6.46.4.24 __STATIC_INLINE void Chip_SYSCON_SetSYSTCKCAL ( uint32_t *sysCalVal* )

Set System tick timer calibration value.

**Parameters**

| | |
|---|---|
| *sysCalVal* | : System tick timer calibration value |

**Returns**

Nothing

Definition at line 207 of file syscon_5411x.h.

#### 6.46.4.25 __STATIC_INLINE void Chip_SYSCON_SetUSARTFRGCtrl ( uint8_t *fmul,* uint8_t *fdiv* )

Set UART Fractional divider value.

**Parameters**

| | |
|---|---|
| *fmul* | : Fractional multiplier value |
| *fdiv* | : Fractional divider value (Must always be 0xFF) |

**Returns**

Nothing

Definition at line 254 of file syscon_5411x.h.

#### 6.46.4.26 __STATIC_INLINE void Chip_SYSCON_StartFreqMeas ( void )

Starts a frequency measurement cycle.

**Returns**

Nothing

**Note**

This function is meant to be used with the Chip_INMUX_SetFreqMeasRefClock() and Chip_INMUX_SetFreqMeasTargClock() functions.

Definition at line 387 of file syscon_5411x.h.

## 6.47 CHIP: LPC5411X UART Driver

### 6.47.1 Detailed Description

### Data Structures

- struct LPC_USART_T

  *UART Registers.*
- struct UART_BAUD_T

  *UART Baud rate calculation structure.*
- struct UART_STATISTICS_T

  *UART statistics structure.*

### Macros

- #define ECHO_EN 1
- #define ECHO_DIS 0
- #define UART_CFG_BITMASK (0x00fddbfd)

  *UART CFG register definitions.*
- #define UART_CFG_ENABLE (0x01 << 0)
- #define UART_CFG_DATALEN_7 (0x00 << 2)
- #define UART_CFG_DATALEN_8 (0x01 << 2)
- #define UART_CFG_DATALEN_9 (0x02 << 2)
- #define UART_CFG_PARITY_NONE (0x00 << 4)
- #define UART_CFG_PARITY_EVEN (0x02 << 4)
- #define UART_CFG_PARITY_ODD (0x03 << 4)
- #define UART_CFG_STOPLEN_1 (0x00 << 6)
- #define UART_CFG_STOPLEN_2 (0x01 << 6)
- #define UART_CFG_MODE32K (0x01 << 7)
- #define UART_CFG_LINMODE (0x01 << 8)
- #define UART_CFG_CTSEN (0x01 << 9)
- #define UART_CFG_SYNCEN (0x01 << 11)
- #define UART_CFG_CLKPOL (0x01 << 12)
- #define UART_CFG_SYNCMST (0x01 << 14)
- #define UART_CFG_LOOP (0x01 << 15)
- #define UART_CFG_IOMODE (0x01 << 16)
- #define UART_CFG_OETA (0x01 << 18)
- #define UART_CFG_AUTOADDR (0x01 << 19)
- #define UART_CFG_OESEL (0x01 << 20)
- #define UART_CFG_OEPOL (0x01 << 21)
- #define UART_CFG_RXPOL (0x01 << 22)
- #define UART_CFG_TXPOL (0x01 << 23)
- #define UART_CTRL_TXBRKEN (0x01 << 1)

  *UART CTRL register definitions.*
- #define UART_CTRL_ADDRDET (0x01 << 2)
- #define UART_CTRL_TXDIS (0x01 << 6)
- #define UART_CTRL_CC (0x01 << 8)
- #define UART_CTRL_CLRCCONRX (0x01 << 9)
- #define UART_CTRL_AUTOBAUD (0x01 << 16)
- #define UART_STAT_RXIDLE (0x01 << 1)

  *UART STAT register definitions.*
- #define UART_STAT_TXIDLE (0x01 << 3)
- #define UART_STAT_CTS (0x01 << 4)

- #define UART_STAT_DELTACTS (0x01 << 5)
- #define UART_STAT_TXDISINT (0x01 << 6)
- #define UART_STAT_RXBRK (0x01 << 10)
- #define UART_STAT_DELTARXBRK (0x01 << 11)
- #define UART_STAT_START (0x01 << 12)
- #define UART_STAT_FRM_ERRINT (0x01 << 13)
- #define UART_STAT_PAR_ERRINT (0x01 << 14)
- #define UART_STAT_RXNOISEINT (0x01 << 15)
- #define UART_STAT_ABERR (0x01 << 16)
- #define UART_INT_TXIDLE (0x01 << 3)

    *UART INTENSET/INTENCLR/INTSTAT register definitions.*

- #define UART_INT_DELTACTS (0x01 << 5)
- #define UART_INT_TXDIS (0x01 << 6)
- #define UART_INT_DELTARXBRK (0x01 << 11)
- #define UART_INT_START (0x01 << 12)
- #define UART_INT_FRAMERR (0x01 << 13)
- #define UART_INT_PARITYERR (0x01 << 14)
- #define UART_INT_RXNOISE (0x01 << 15)
- #define UART_INT_ABERR (0x01 << 16)
- #define UART_FIFOCFG_BITMASK (0x7F033)

    *UART FIFO Configuration register bits.*

- #define UART_FIFOCFG_ENABLETX (1 << 0)
- #define UART_FIFOCFG_ENABLERX (1 << 1)
- #define UART_FIFOCFG_DMATX (1 << 12)
- #define UART_FIFOCFG_DMARX (1 << 13)
- #define UART_FIFOCFG_WAKETX (1 << 14)
- #define UART_FIFOCFG_WAKERX (1 << 15)
- #define UART_FIFOCFG_EMPTYTX (1 << 16)
- #define UART_FIFOCFG_EMPTYRX (1 << 17)
- #define UART_FIFO_DEPTH (16) /** UART-FIFO How many entries are in the FIFO */

    *UART FIFO Status register defines.*

- #define UART_FIFOSTAT_BITMASK (0x1F1FFB) /** UART-FIFO STAT Register BitMask */
- #define UART_FIFOSTAT_TXERR (1 << 0)
- #define UART_FIFOSTAT_RXERR (1 << 1)
- #define UART_FIFOSTAT_PERIPH (1 << 3)
- #define UART_FIFOSTAT_TXEMPTY (1 << 4)
- #define UART_FIFOSTAT_TXNOTFULL (1 << 5)
- #define UART_FIFOSTAT_RXNOTEMPTY (1 << 6)
- #define UART_FIFOSTAT_RXFULL (1 << 7)
- #define UART_FIFOSTAT_TXLVL(lvl) (((lvl) >> 8) & 0x1F)
- #define UART_FIFOSTAT_RXLVL(lvl) (((lvl) >> 16) & 0x1F)
- #define UART_FIFOTRIG_BITMASK (0x000f0f03) /** UART FIFO trigger settings Register BitMask */

    *UART FIFO trigger settings register defines.*

- #define UART_FIFOTRIG_TXLVLENA (1 << 0)
- #define UART_FIFOTRIG_RXLVLENA (1 << 1)
- #define UART_FIFOTRIG_TXLVL(lvl) ((lvl & 0x0f) << 8)
- #define UART_FIFOTRIG_RXLVL(lvl) ((lvl & 0x0f) << 16)
- #define UART_FIFOINT_BITMASK (0x001F) /** FIFO interrupt Bit mask */

    *UART FIFO Interrupt enable/disable/status [INTSET/INTCLR/INTSTAT and FIFOINTSET/FIFOINTCLR/FIFOINTSTAT registers].*

- #define UART_FIFOINT_TXERR (1 << 0) /** TX error interrupt [BIT-0 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */
- #define UART_FIFOINT_RXERR (1 << 1) /** RX error interrupt [BIT-1 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] */

- #define UART_FIFOINT_TXLVL (1 << 2) /∗∗ TX FIFO ready interrupt [BIT-2 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] ∗/
- #define UART_FIFOINT_RXLVL (1 << 3) /∗∗ RX Data ready interrupt [BIT-3 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] ∗/
- #define UART_FIFOINT_PERINT (1 << 4) /∗∗ UART peripheral interrupt [BIT-4 of FIFOINTSTAT register] ∗/

## Functions

- __STATIC_INLINE void Chip_UART_Enable (LPC_USART_T ∗pUART)

  *Enable the UART.*
- __STATIC_INLINE void Chip_UART_Disable (LPC_USART_T ∗pUART)

  *Disable the UART.*
- __STATIC_INLINE void Chip_UART_TXEnable (LPC_USART_T ∗pUART)

  *Enable transmission on UART TxD pin.*
- __STATIC_INLINE void Chip_UART_TXDisable (LPC_USART_T ∗pUART)

  *Disable transmission on UART TxD pin.*
- __STATIC_INLINE uint32_t Chip_UART_AutoBaud (LPC_USART_T ∗pUART)

  *Set auto baud.*
- __STATIC_INLINE void Chip_UART_SendByte (LPC_USART_T ∗pUART, uint8_t data)

  *Transmit a single data byte through the UART peripheral.*
- __STATIC_INLINE uint32_t Chip_UART_ReadByte (LPC_USART_T ∗pUART)

  *Read a single byte data from the UART peripheral.*
- __STATIC_INLINE void Chip_UART_IntEnable (LPC_USART_T ∗pUART, uint32_t intMask)

  *Enable UART interrupts.*
- __STATIC_INLINE void Chip_UART_IntDisable (LPC_USART_T ∗pUART, uint32_t intMask)

  *Disable UART interrupts.*
- __STATIC_INLINE uint32_t Chip_UART_GetIntsEnabled (LPC_USART_T ∗pUART)

  *Returns UART interrupts that are enabled.*
- __STATIC_INLINE uint32_t Chip_UART_GetIntStatus (LPC_USART_T ∗pUART)

  *Get UART interrupt status.*
- __STATIC_INLINE void Chip_UART_ConfigData (LPC_USART_T ∗pUART, uint32_t config)

  *Configure data width, parity and stop bits.*
- __STATIC_INLINE uint32_t Chip_UART_GetStatus (LPC_USART_T ∗pUART)

  *Get the UART status register.*
- __STATIC_INLINE void Chip_UART_ClearStatus (LPC_USART_T ∗pUART, uint32_t stsMask)

  *Clear the UART status register.*
- __STATIC_INLINE uint32_t Chip_UART_GetFIFOStatus (LPC_USART_T ∗pUART)

  *Get the current status of UART controller FIFO.*
- __STATIC_INLINE void Chip_UART_ClearFIFOStatus (LPC_USART_T ∗pUART, uint32_t mask)

  *Clear the FIFO status register.*
- __STATIC_INLINE void Chip_UART_SetFIFOTrigLevel (LPC_USART_T ∗pUART, uint8_t tx_lvl, uint8_t rx_lvl)

  *Setup the trigger level for UART FIFO.*
- __STATIC_INLINE void Chip_UART_EnableFIFOInts (LPC_USART_T ∗pUART, uint32_t intMask)

  *Enable a UART FIFO interrupt.*
- __STATIC_INLINE void Chip_UART_DisableFIFOInts (LPC_USART_T ∗pUART, uint32_t intMask)

  *Disable a UART FIFO interrupt.*
- __STATIC_INLINE uint32_t Chip_UART_GetFIFOEnabledInts (LPC_USART_T ∗pUART)

  *Return enabled UART FIFO interrupts.*
- __STATIC_INLINE uint32_t Chip_UART_GetFIFOPendingInts (LPC_USART_T ∗pUART)

*Return pending UART FIFO interrupts.*

- __STATIC_INLINE void Chip_UART_SetFIFOCfg (LPC_USART_T ∗pUART, uint32_t cfg)

  *Set FIFO Configuration register.*
- __STATIC_INLINE void Chip_UART_ClearFIFOCfg (LPC_USART_T ∗pUART, uint32_t cfg)

  *Clear FIFO Configuration register.*
- __STATIC_INLINE void Chip_UART_FlushFIFOs (LPC_USART_T ∗pUART)

  *Flush FIFOs.*
- int Chip_UART_Init (LPC_USART_T ∗pUART)

  *Initialize the UART peripheral.*
- void Chip_UART_DeInit (LPC_USART_T ∗pUART)

  *Deinitialize the UART peripheral.*
- void Chip_UART_ConfigDMA (LPC_USART_T ∗pUART)

  *Configure UART for DMA.*
- int Chip_UART_Send (LPC_USART_T ∗pUART, const void ∗data, int numBytes)

  *Transmit a byte array through the UART peripheral (non-blocking)*
- int Chip_UART_Read (LPC_USART_T ∗pUART, void ∗data, int numBytes)

  *Read data through the UART peripheral (non-blocking)*
- uint32_t Chip_UART_SetBaud (LPC_USART_T ∗pUART, uint32_t baudrate)

  *Set baud rate for UART.*
- int Chip_UART_SendBlocking (LPC_USART_T ∗pUART, const void ∗data, int numBytes)

  *Transmit a byte array through the UART peripheral (blocking)*
- int Chip_UART_ReadBlocking (LPC_USART_T ∗pUART, void ∗data, int numBytes)

  *Read data through the UART peripheral (blocking)*
- void Chip_UART_RXIntHandlerRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB)

  *UART receive-only interrupt handler for ring buffers.*
- void Chip_UART_TXIntHandlerRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB)

  *UART transmit-only interrupt handler for ring buffers.*
- uint32_t Chip_UART_SendRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB, const void ∗data, int count)

  *Populate a transmit ring buffer and start UART transmit.*
- int Chip_UART_ReadRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB, void ∗data, int bytes)

  *Copy data from a receive ring buffer.*
- void Chip_UART_IRQHandlerRB (LPC_USART_T ∗pUART, UART_STATISTICS_T ∗statistics, RINGBUFF_T ∗pRXRB, RINGBUFF_T ∗pTXRB)

  *UART receive/transmit interrupt handler for ring buffers.*
- void Chip_UART_IRQHandlerDMA (LPC_USART_T ∗pUART, UART_STATISTICS_T ∗statistics)

  *UART receive/transmit interrupt handler for DMA.*

## 6.47.2 Macro Definition Documentation

### 6.47.2.1 #define ECHO_DIS 0

Definition at line 50 of file uart_5411x.h.

### 6.47.2.2 #define ECHO_EN 1

Definition at line 49 of file uart_5411x.h.

### 6.47.2.3 #define UART_CFG_AUTOADDR (0x01 << 19)

Automatic Address matching enable

Definition at line 109 of file uart_5411x.h.

#### 6.47.2.4 #define UART_CFG_BITMASK (0x00fddbfd)

UART CFG register definitions.

Definition at line 90 of file uart_5411x.h.

#### 6.47.2.5 #define UART_CFG_CLKPOL (0x01 << 12)

Un_RXD rising edge sample enable bit

Definition at line 104 of file uart_5411x.h.

#### 6.47.2.6 #define UART_CFG_CTSEN (0x01 << 9)

CTS enable bit

Definition at line 102 of file uart_5411x.h.

#### 6.47.2.7 #define UART_CFG_DATALEN_7 (0x00 << 2)

UART 7 bit length mode

Definition at line 92 of file uart_5411x.h.

#### 6.47.2.8 #define UART_CFG_DATALEN_8 (0x01 << 2)

UART 8 bit length mode

Definition at line 93 of file uart_5411x.h.

#### 6.47.2.9 #define UART_CFG_DATALEN_9 (0x02 << 2)

UART 9 bit length mode

Definition at line 94 of file uart_5411x.h.

#### 6.47.2.10 #define UART_CFG_ENABLE (0x01 << 0)

Definition at line 91 of file uart_5411x.h.

#### 6.47.2.11 #define UART_CFG_IOMODE (0x01 << 16)

enable bit standard UART / IrDA UART

Definition at line 107 of file uart_5411x.h.

#### 6.47.2.12 #define UART_CFG_LINMODE (0x01 << 8)

UART LIN MODE

Definition at line 101 of file uart_5411x.h.

#### 6.47.2.13 #define UART_CFG_LOOP (0x01 << 15)

Loopback mode enable bit

Definition at line 106 of file uart_5411x.h.

#### 6.47.2.14 #define UART_CFG_MODE32K (0x01 << 7)

UART 32K MODE

Definition at line 100 of file uart_5411x.h.

#### 6.47.2.15 #define UART_CFG_OEPOL (0x01 << 21)

Output Enable Polarity

Definition at line 111 of file uart_5411x.h.

#### 6.47.2.16 #define UART_CFG_OESEL (0x01 << 20)

Output Enable Select

Definition at line 110 of file uart_5411x.h.

#### 6.47.2.17 #define UART_CFG_OETA (0x01 << 18)

Output Enable Turnaround time enable for RS-485 operation

Definition at line 108 of file uart_5411x.h.

#### 6.47.2.18 #define UART_CFG_PARITY_EVEN (0x02 << 4)

Even parity

Definition at line 96 of file uart_5411x.h.

#### 6.47.2.19 #define UART_CFG_PARITY_NONE (0x00 << 4)

No parity

Definition at line 95 of file uart_5411x.h.

#### 6.47.2.20 #define UART_CFG_PARITY_ODD (0x03 << 4)

Odd parity

Definition at line 97 of file uart_5411x.h.

#### 6.47.2.21 #define UART_CFG_RXPOL (0x01 << 22)

Receive data polarity

Definition at line 112 of file uart_5411x.h.

#### 6.47.2.22 #define UART_CFG_STOPLEN_1 (0x00 << 6)

UART One Stop Bit Select

Definition at line 98 of file uart_5411x.h.

#### 6.47.2.23 #define UART_CFG_STOPLEN_2 (0x01 << 6)

UART Two Stop Bits Select

Definition at line 99 of file uart_5411x.h.

#### 6.47.2.24 #define UART_CFG_SYNCEN (0x01 << 11)

Synchronous mode enable bit

Definition at line 103 of file uart_5411x.h.

#### 6.47.2.25 #define UART_CFG_SYNCMST (0x01 << 14)

Select master mode (synchronous mode) enable bit

Definition at line 105 of file uart_5411x.h.

#### 6.47.2.26 #define UART_CFG_TXPOL (0x01 << 23)

Transmit data polarity

Definition at line 113 of file uart_5411x.h.

#### 6.47.2.27 #define UART_CTRL_ADDRDET (0x01 << 2)

Address detect mode enable bit

Definition at line 120 of file uart_5411x.h.

#### 6.47.2.28 #define UART_CTRL_AUTOBAUD (0x01 << 16)

Auto baud bit

Definition at line 124 of file uart_5411x.h.

#### 6.47.2.29 #define UART_CTRL_CC (0x01 << 8)

Continuous Clock mode enable bit

Definition at line 122 of file uart_5411x.h.

#### 6.47.2.30 #define UART_CTRL_CLRCCONRX (0x01 << 9)

Clear Continuous Clock bit

Definition at line 123 of file uart_5411x.h.

#### 6.47.2.31 #define UART_CTRL_TXBRKEN (0x01 $<<$ 1)

UART CTRL register definitions.

Continuous break enable bit

Definition at line 119 of file uart_5411x.h.

#### 6.47.2.32 #define UART_CTRL_TXDIS (0x01 $<<$ 6)

Transmit disable bit

Definition at line 121 of file uart_5411x.h.

#### 6.47.2.33 #define UART_FIFO_DEPTH (16) /** UART-FIFO How many entries are in the FIFO */

UART FIFO Status register defines.

Definition at line 171 of file uart_5411x.h.

#### 6.47.2.34 #define UART_FIFOCFG_BITMASK (0x7F033)

UART FIFO Configuration register bits.

Register mask bit

Definition at line 158 of file uart_5411x.h.

#### 6.47.2.35 #define UART_FIFOCFG_DMARX (1 $<<$ 13)

Enable DMA RX

Definition at line 162 of file uart_5411x.h.

#### 6.47.2.36 #define UART_FIFOCFG_DMATX (1 $<<$ 12)

Enable DMA TX

Definition at line 161 of file uart_5411x.h.

#### 6.47.2.37 #define UART_FIFOCFG_EMPTYRX (1 $<<$ 17)

Empty the RX FIFO

Definition at line 166 of file uart_5411x.h.

#### 6.47.2.38 #define UART_FIFOCFG_EMPTYTX (1 $<<$ 16)

Empty the TX FIFO

Definition at line 165 of file uart_5411x.h.

#### 6.47.2.39 #define UART_FIFOCFG_ENABLERX (1 $<<$ 1)

Enable RX FIFO

Definition at line 160 of file uart_5411x.h.

#### 6.47.2.40 #define UART_FIFOCFG_ENABLETX (1 $<<$ 0)

Enable TX FIFO

Definition at line 159 of file uart_5411x.h.

#### 6.47.2.41 #define UART_FIFOCFG_WAKERX (1 $<<$ 15)

Enable wakeup triggered by RX

Definition at line 164 of file uart_5411x.h.

#### 6.47.2.42 #define UART_FIFOCFG_WAKETX (1 $<<$ 14)

Enable wakeup triggered by TX

Definition at line 163 of file uart_5411x.h.

#### 6.47.2.43 #define UART_FIFOINT_BITMASK (0x001F) /$**$ FIFO interrupt Bit mask $*$/

UART FIFO Interrupt enable/disable/status [INTSET/INTCLR/INTSTAT and FIFOINTSET/FIFOINTCLR/FIFOIN↩ TSTAT registers].

Definition at line 195 of file uart_5411x.h.

#### 6.47.2.44 #define UART_FIFOINT_PERINT (1 $<<$ 4) /$**$ UART peripheral interrupt [BIT-4 of FIFOINTSTAT register] $*$/

Definition at line 200 of file uart_5411x.h.

#### 6.47.2.45 #define UART_FIFOINT_RXERR (1 $<<$ 1) /$**$ RX error interrupt [BIT-1 of FIFOINTENSET/FIFOINTENCLR/FIFOINTST↩ AT register] $*$/

Definition at line 197 of file uart_5411x.h.

#### 6.47.2.46 #define UART_FIFOINT_RXLVL (1 $<<$ 3) /$**$ RX Data ready interrupt [BIT-3 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] $*$/

Definition at line 199 of file uart_5411x.h.

#### 6.47.2.47 #define UART_FIFOINT_TXERR (1 $<<$ 0) /$**$ TX error interrupt [BIT-0 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] $*$/

Definition at line 196 of file uart_5411x.h.

#### 6.47.2.48 #define UART_FIFOINT_TXLVL (1 $<<$ 2) /$**$ TX FIFO ready interrupt [BIT-2 of FIFOINTENSET/FIFOINTENCLR/FIFOINTSTAT register] $*$/

Definition at line 198 of file uart_5411x.h.

#### 6.47.2.49 #define UART_FIFOSTAT_BITMASK (0x1F1FFB) /$**$ UART-FIFO STAT Register BitMask $*$/

Definition at line 172 of file uart_5411x.h.

#### 6.47.2.50 #define UART_FIFOSTAT_PERIPH (1 << 3)

Peripheral interrupt

Definition at line 175 of file uart_5411x.h.

#### 6.47.2.51 #define UART_FIFOSTAT_RXERR (1 << 1)

UART RX Error

Definition at line 174 of file uart_5411x.h.

#### 6.47.2.52 #define UART_FIFOSTAT_RXFULL (1 << 7)

RX FIFO Full

Definition at line 179 of file uart_5411x.h.

#### 6.47.2.53 #define UART_FIFOSTAT_RXLVL( *lvl* ) (((lvl) >> 16) & 0x1F)

Get TX Level from status

Definition at line 181 of file uart_5411x.h.

#### 6.47.2.54 #define UART_FIFOSTAT_RXNOTEMPTY (1 << 6)

RXFIFO Not empty

Definition at line 178 of file uart_5411x.h.

#### 6.47.2.55 #define UART_FIFOSTAT_TXEMPTY (1 << 4)

TXFIFO Empty

Definition at line 176 of file uart_5411x.h.

#### 6.47.2.56 #define UART_FIFOSTAT_TXERR (1 << 0)

UART TX Error

Definition at line 173 of file uart_5411x.h.

#### 6.47.2.57 #define UART_FIFOSTAT_TXLVL( *lvl* ) (((lvl) >> 8) & 0x1F)

Get TX Level from status

Definition at line 180 of file uart_5411x.h.

#### 6.47.2.58 #define UART_FIFOSTAT_TXNOTFULL (1 << 5)

TXFIFO Not Full

Definition at line 177 of file uart_5411x.h.

#### 6.47.2.59 #define UART_FIFOTRIG_BITMASK (0x000f0f03) /** UART FIFO trigger settings Register BitMask */

UART FIFO trigger settings register defines.

Definition at line 186 of file uart_5411x.h.

#### 6.47.2.60 #define UART_FIFOTRIG_RXLVL( *lvl* ) ((lvl & 0x0f) << 16)

Set RX Level trigger

Definition at line 190 of file uart_5411x.h.

#### 6.47.2.61 #define UART_FIFOTRIG_RXLVLENA (1 << 1)

RX level enable

Definition at line 188 of file uart_5411x.h.

#### 6.47.2.62 #define UART_FIFOTRIG_TXLVL( *lvl* ) ((lvl & 0x0f) << 8)

Set TX Level trigger

Definition at line 189 of file uart_5411x.h.

#### 6.47.2.63 #define UART_FIFOTRIG_TXLVLENA (1 << 0)

TX level enable

Definition at line 187 of file uart_5411x.h.

#### 6.47.2.64 #define UART_INT_ABERR (0x01 << 16)

Auto-baud error interrupt

Definition at line 153 of file uart_5411x.h.

#### 6.47.2.65 #define UART_INT_DELTACTS (0x01 << 5)

Change in CTS state interrupt

Definition at line 146 of file uart_5411x.h.

#### 6.47.2.66 #define UART_INT_DELTARXBRK (0x01 << 11)

Change in receiver break detection interrupt

Definition at line 148 of file uart_5411x.h.

#### 6.47.2.67 #define UART_INT_FRAMERR (0x01 << 13)

Frame error interrupt

Definition at line 150 of file uart_5411x.h.

#### 6.47.2.68 #define UART_INT_PARITYERR (0x01 << 14)

Parity error interrupt

Definition at line 151 of file uart_5411x.h.

#### 6.47.2.69 #define UART_INT_RXNOISE (0x01 << 15)

Received noise interrupt

Definition at line 152 of file uart_5411x.h.

#### 6.47.2.70 #define UART_INT_START (0x01 << 12)

Start detect interrupt

Definition at line 149 of file uart_5411x.h.

#### 6.47.2.71 #define UART_INT_TXDIS (0x01 << 6)

Transmitter disable interrupt

Definition at line 147 of file uart_5411x.h.

#### 6.47.2.72 #define UART_INT_TXIDLE (0x01 << 3)

UART INTENSET/INTENCLR/INTSTAT register definitions.

Transmit idle interrupt

Definition at line 145 of file uart_5411x.h.

#### 6.47.2.73 #define UART_STAT_ABERR (0x01 << 16)

Auto baud error flag

Definition at line 140 of file uart_5411x.h.

#### 6.47.2.74 #define UART_STAT_CTS (0x01 << 4)

Status of CTS signal

Definition at line 131 of file uart_5411x.h.

#### 6.47.2.75 #define UART_STAT_DELTACTS (0x01 << 5)

Change in CTS state

Definition at line 132 of file uart_5411x.h.

#### 6.47.2.76 #define UART_STAT_DELTARXBRK (0x01 << 11)

Change in receive break detection

Definition at line 135 of file uart_5411x.h.

#### 6.47.2.77 #define UART_STAT_FRM_ERRINT (0x01 << 13)

Framing Error interrupt flag

Definition at line 137 of file uart_5411x.h.

#### 6.47.2.78 #define UART_STAT_PAR_ERRINT (0x01 << 14)

Parity Error interrupt flag

Definition at line 138 of file uart_5411x.h.

#### 6.47.2.79 #define UART_STAT_RXBRK (0x01 << 10)

Received break

Definition at line 134 of file uart_5411x.h.

#### 6.47.2.80 #define UART_STAT_RXIDLE (0x01 << 1)

UART STAT register definitions.

Receiver idle

Definition at line 129 of file uart_5411x.h.

#### 6.47.2.81 #define UART_STAT_RXNOISEINT (0x01 << 15)

Received Noise interrupt flag

Definition at line 139 of file uart_5411x.h.

#### 6.47.2.82 #define UART_STAT_START (0x01 << 12)

Start detected

Definition at line 136 of file uart_5411x.h.

#### 6.47.2.83 #define UART_STAT_TXDISINT (0x01 << 6)

Transmitter disabled

Definition at line 133 of file uart_5411x.h.

#### 6.47.2.84 #define UART_STAT_TXIDLE (0x01 << 3)

Transmitter idle

Definition at line 130 of file uart_5411x.h.

### 6.47.3 Function Documentation

#### 6.47.3.1 __STATIC_INLINE uint32_t Chip_UART_AutoBaud ( LPC_USART_T * *pUART* )

Set auto baud.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected pUART peripheral |

**Returns**

true if auto baud succeeds, false if fails

Definition at line 284 of file uart_5411x.h.

#### 6.47.3.2 __STATIC_INLINE void Chip_UART_ClearFIFOCfg ( LPC_USART_T ∗ *pUART,* uint32_t *cfg* )

Clear FIFO Configuration register.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |
| *cfg* | : Configuration value mask (OR'ed UART_FIFOCFG_∗ values like UART_FIFOCFG_ENABLETX) |

**Returns**

Nothing

Definition at line 516 of file uart_5411x.h.

#### 6.47.3.3 __STATIC_INLINE void Chip_UART_ClearFIFOStatus ( LPC_USART_T ∗ *pUART,* uint32_t *mask* )

Clear the FIFO status register.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of the UART peripheral on the chip |
| *mask* | : Mask of the status bits that needs to be cleared |

**Returns**

Nothing

Definition at line 436 of file uart_5411x.h.

#### 6.47.3.4 __STATIC_INLINE void Chip_UART_ClearStatus ( LPC_USART_T ∗ *pUART,* uint32_t *stsMask* )

Clear the UART status register.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UARTx peripheral |
| *stsMask* | : OR'ed statuses to disable |

**Returns**

Nothing

**Note**

Multiple interrupts may be pending. Mask the return value with one or more UART_INTEN_∗ definitions to determine pending interrupts.

Definition at line 413 of file uart_5411x.h.

#### 6.47.3.5 __STATIC_INLINE void Chip_UART_ConfigData ( LPC_USART_T * *pUART,* uint32_t *config* )

Configure data width, parity and stop bits.

**Parameters**

|  |  |
|---|---|
| *pUART* | : Pointer to selected pUART peripheral |
| *config* | : UART configuration, OR'ed values of select UART_CFG_∗ defines |

**Returns**

Nothing

**Note**

Select OR'ed config options for the UART from the UART_CFG_PARITY_∗, UART_CFG_STOPLEN_∗, and UART_CFG_DATALEN_∗ definitions. For example, a configuration of 8 data bits, 1 stop bit, and even (enabled) parity would be (UART_CFG_DATALEN_8 | UART_CFG_STOPLEN_1 | UART_CFG_PARITY_EV↩ EN). Will not alter other bits in the CFG register.

Definition at line 386 of file uart_5411x.h.

#### 6.47.3.6 void Chip_UART_ConfigDMA ( LPC_USART_T ∗ *pUART* )

Configure UART for DMA.

**Parameters**

|  |  |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

Nothing

Definition at line 174 of file uart_5411x.c.

#### 6.47.3.7 void Chip_UART_DeInit ( LPC_USART_T ∗ *pUART* )

Deinitialize the UART peripheral.

**Parameters**

|  |  |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

Nothing

Definition at line 168 of file uart_5411x.c.

#### 6.47.3.8 __STATIC_INLINE void Chip_UART_Disable ( LPC_USART_T ∗ *pUART* )

Disable the UART.

**Parameters**

|  |  |
|---|---|
| *pUART* | : Pointer to selected UARTx peripheral |

**Returns**

Nothing

Definition at line 254 of file uart_5411x.h.

#### 6.47.3.9 __STATIC_INLINE void Chip_UART_DisableFIFOInts ( LPC_USART_T ∗ *pUART,* uint32_t *intMask* )

Disable a UART FIFO interrupt.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |
| *intMask* | : Or'ed value of UART_FIFOINT_∗ values to disable (See UART_FIFOINT_BITMASK) |

**Returns**

Nothing

Definition at line 474 of file uart_5411x.h.

#### 6.47.3.10 __STATIC_INLINE void Chip_UART_Enable ( LPC_USART_T ∗ *pUART* )

Enable the UART.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UARTx peripheral |

**Returns**

Nothing

Definition at line 244 of file uart_5411x.h.

#### 6.47.3.11 __STATIC_INLINE void Chip_UART_EnableFIFOInts ( LPC_USART_T ∗ *pUART,* uint32_t *intMask* )

Enable a UART FIFO interrupt.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |
| *intMask* | : Or'ed value of UART_FIFOINT_∗ values to enable |

**Returns**

Nothing

Definition at line 463 of file uart_5411x.h.

#### 6.47.3.12 __STATIC_INLINE void Chip_UART_FlushFIFOs ( LPC_USART_T ∗ *pUART* )

Flush FIFOs.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

Nothing

Definition at line 526 of file uart_5411x.h.

#### 6.47.3.13 __STATIC_INLINE uint32_t Chip_UART_GetFIFOEnabledInts ( LPC_USART_T ∗ *pUART* )

Return enabled UART FIFO interrupts.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

An Or'ed value of UART_FIFOINT_∗ values

Definition at line 484 of file uart_5411x.h.

#### 6.47.3.14 __STATIC_INLINE uint32_t Chip_UART_GetFIFOPendingInts ( LPC_USART_T ∗ *pUART* )

Return pending UART FIFO interrupts.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

An Or'ed value of UART_FIFOINT_∗ values

Definition at line 494 of file uart_5411x.h.

#### 6.47.3.15 __STATIC_INLINE uint32_t Chip_UART_GetFIFOStatus ( LPC_USART_T ∗ *pUART* )

Get the current status of UART controller FIFO.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

UART Status (Or-ed bit value of UART_FIFOSTAT_∗)

**Note**

Mask the return value with a value of type UART_FIFOSTAT_∗ to determine if that status is active.

Definition at line 425 of file uart_5411x.h.

#### 6.47.3.16 __STATIC_INLINE uint32_t Chip_UART_GetIntsEnabled ( LPC_USART_T ∗ *pUART* )

Returns UART interrupts that are enabled.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |

**Returns**

Returns the enabled UART interrupts

**Note**

Use an OR'ed value of UART_INTEN_∗ definitions with this function to determine which interrupts are enabled. You can check for multiple enabled bits if needed.

Definition at line 357 of file uart_5411x.h.

#### 6.47.3.17 __STATIC_INLINE uint32_t Chip_UART_GetIntStatus ( LPC_USART_T ∗ *pUART* )

Get UART interrupt status.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

The UART interrupt status register

**Note**

This register does not indicate UART pending interrupts. It indicates the enabled interrupts. To get pending interrupts, read the UART status register.

Definition at line 370 of file uart_5411x.h.

### 6.47.3.18 __STATIC_INLINE uint32_t Chip_UART_GetStatus ( LPC_USART_T ∗ *pUART* )

Get the UART status register.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UARTx peripheral |

**Returns**

UART status register

**Note**

Multiple statuses may be pending. Mask the return value with one or more UART_STAT_∗ definitions to determine statuses.

Definition at line 399 of file uart_5411x.h.

### 6.47.3.19 int Chip_UART_Init ( LPC_USART_T ∗ *pUART* )

Initialize the UART peripheral.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |

**Returns**

0 - On success; ERR_FLEXCOMM_FUNCNOTSUPPORTED or ERR_FLEXCOMM_NOTFREE on failure

Definition at line 155 of file uart_5411x.c.

### 6.47.3.20 __STATIC_INLINE void Chip_UART_IntDisable ( LPC_USART_T ∗ *pUART,* uint32_t *intMask* )

Disable UART interrupts.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *intMask* | : OR'ed Interrupts to disable |

**Returns**

Nothing

**Note**

Use an OR'ed value of UART_INTEN_∗ definitions with this function to disable specific UART interrupts.

Definition at line 344 of file uart_5411x.h.

#### 6.47.3.21 __STATIC_INLINE void Chip_UART_IntEnable ( LPC_USART_T ∗ *pUART,* uint32_t *intMask* )

Enable UART interrupts.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *intMask* | : OR'ed Interrupts to enable |

**Returns**

Nothing

**Note**

Use an OR'ed value of UART_INTEN_∗ definitions with this function to enable specific UART interrupts.

Definition at line 331 of file uart_5411x.h.

#### 6.47.3.22 void Chip_UART_IRQHandlerDMA ( LPC_USART_T ∗ *pUART,* UART_STATISTICS_T ∗ *statistics* )

UART receive/transmit interrupt handler for DMA.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *statistics* | : Pointer to statistics structure |

**Returns**

Nothing

**Note**

This interrupt service routine provides a basic implementation of the UART interrupt handler for DMA

Definition at line 387 of file uart_5411x.c.

#### 6.47.3.23 void Chip_UART_IRQHandlerRB ( LPC_USART_T ∗ *pUART,* UART_STATISTICS_T ∗ *statistics,* RINGBUFF_T ∗ *pRXRB,* RINGBUFF_T ∗ *pTXRB* )

UART receive/transmit interrupt handler for ring buffers.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *statistics* | : Pointer to statistics structure |
| *pRXRB* | : Pointer to transmit ring buffer |
| *pTXRB* | : Pointer to receive ring buffer |

**Returns**

Nothing

**Note**

This provides a basic implementation of the UART IRQ handler for support of a ring buffer implementation for transmit and receive.

Definition at line 327 of file uart_5411x.c.

#### 6.47.3.24 int Chip_UART_Read ( LPC_USART_T ∗ *pUART,* void ∗ *data,* int *numBytes* )

Read data through the UART peripheral (non-blocking)

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *data* | : Pointer to bytes array to fill |
| *numBytes* | : Size of the passed data array |

**Returns**

The actual number of bytes read

**Note**

This function reads data from the receive FIFO until either all the data has been read or the passed buffer is completely full. This function will not block. This function ignores errors.

Definition at line 211 of file uart_5411x.c.

#### 6.47.3.25 int Chip_UART_ReadBlocking ( LPC_USART_T ∗ *pUART,* void ∗ *data,* int *numBytes* )

Read data through the UART peripheral (blocking)

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *data* | : Pointer to data array to fill |
| *numBytes* | : Size of the passed data array |

**Returns**

The size of the dat array

**Note**

This function reads data from the receive FIFO until the passed buffer is completely full. The function will block until full. This function ignores errors.

Definition at line 226 of file uart_5411x.c.

#### 6.47.3.26 __STATIC_INLINE uint32_t Chip_UART_ReadByte ( LPC_USART_T ∗ *pUART* )

Read a single byte data from the UART peripheral.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |

**Returns**

A single byte of data read

**Note**

This function reads a byte from the UART receive FIFO or receive hold register regard regardless of UART state.

Definition at line 317 of file uart_5411x.h.

#### 6.47.3.27 int Chip_UART_ReadRB ( LPC_USART_T ∗ *pUART,* RINGBUFF_T ∗ *pRB,* void ∗ *data,* int *bytes* )

Copy data from a receive ring buffer.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *pRB* | : Pointer to ring buffer structure to use |
| *data* | : Pointer to buffer to fill from ring buffer |
| *bytes* | : Size of the passed buffer in bytes |

**Returns**

The number of bytes placed into the ring buffer

**Note**

Will move the data from the RX ring buffer up to the the maximum passed buffer size. Returns 0 if there is no data in the ring buffer.

Definition at line 319 of file uart_5411x.c.

#### 6.47.3.28 void Chip_UART_RXIntHandlerRB ( LPC_USART_T ∗ *pUART,* RINGBUFF_T ∗ *pRB* )

UART receive-only interrupt handler for ring buffers.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *pRB* | : Pointer to ring buffer structure to use |

**Returns**

Nothing

**Note**

If ring buffer support is desired for the receive side of data transfer, the UART interrupt should call this function for a receive based interrupt status.

Definition at line 279 of file uart_5411x.c.

#### 6.47.3.29 int Chip_UART_Send ( LPC_USART_T ∗ *pUART,* const void ∗ *data,* int *numBytes* )

Transmit a byte array through the UART peripheral (non-blocking)

**Parameters**

| | |
|---:|---|
| *pUART* | : Pointer to selected UART peripheral |
| *data* | : Pointer to bytes to transmit |
| *numBytes* | : Number of bytes to transmit |

**Returns**

The actual number of bytes placed into the FIFO

**Note**

This function places data into the transmit FIFO until either all the data is in the FIFO or the FIFO is full. This function will not block in the FIFO is full. The actual number of bytes placed into the FIFO is returned. This function ignores errors.

Definition at line 180 of file uart_5411x.c.

#### 6.47.3.30 int Chip_UART_SendBlocking ( LPC_USART_T ∗ *pUART,* const void ∗ *data,* int *numBytes* )

Transmit a byte array through the UART peripheral (blocking)

**Parameters**

| | |
|---:|---|
| *pUART* | : Pointer to selected UART peripheral |
| *data* | : Pointer to data to transmit |
| *numBytes* | : Number of bytes to transmit |

**Returns**

The number of bytes transmitted

**Note**

This function will send or place all bytes into the transmit FIFO. This function will block until the last bytes are in the FIFO.

Definition at line 195 of file uart_5411x.c.

#### 6.47.3.31 __STATIC_INLINE void Chip_UART_SendByte ( LPC_USART_T ∗ *pUART,* uint8_t *data* )

Transmit a single data byte through the UART peripheral.

**Parameters**

| | |
|---:|---|
| *pUART* | : Pointer to selected UART peripheral |
| *data* | : Byte to transmit |

**Returns**

Nothing

**Note**

This function attempts to place a byte into the UART transmit holding register regard regardless of UART state.

Definition at line 305 of file uart_5411x.h.

#### 6.47.3.32 uint32_t Chip_UART_SendRB ( LPC_USART_T ∗ *pUART,* RINGBUFF_T ∗ *pRB,* const void ∗ *data,* int *count* )

Populate a transmit ring buffer and start UART transmit.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |
| *pRB* | : Pointer to ring buffer structure to use |
| *data* | : Pointer to buffer to move to ring buffer |
| *count* | : Number of bytes to move |

**Returns**

The number of bytes placed into the ring buffer

**Note**

Will move the data into the TX ring buffer and start the transfer. If the number of bytes returned is less than the number of bytes to send, the ring buffer is considered full.

Definition at line 300 of file uart_5411x.c.

#### 6.47.3.33 uint32_t Chip_UART_SetBaud ( LPC_USART_T ∗ *pUART,* uint32_t *baudrate* )

Set baud rate for UART.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |
| *baudrate* | Baud rate to be set |

**Returns**

Actual baud rate set

**Note**

This function uses the FRG to generate the required clock, the FRG is shared by all the UARTs, hence calling this API for multiple UARTs can affect each other.

Definition at line 252 of file uart_5411x.c.

#### 6.47.3.34 __STATIC_INLINE void Chip_UART_SetFIFOCfg ( LPC_USART_T ∗ *pUART,* uint32_t *cfg* )

Set FIFO Configuration register.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of UART peripheral on the chip |
| *cfg* | : Configuration value mask (OR'ed UART_FIFOCFG_∗ values like UART_FIFOCFG_ENABLETX) |

**Returns**

Nothing

Definition at line 505 of file uart_5411x.h.

#### 6.47.3.35 __STATIC_INLINE void Chip_UART_SetFIFOTrigLevel ( LPC_USART_T ∗ *pUART,* uint8_t *tx_lvl,* uint8_t *rx_lvl* )

Setup the trigger level for UART FIFO.

**Parameters**

| | |
|---|---|
| *pUART* | : The base of the UART peripheral on chip |
| *tx_lvl* | : TX Trigger level [Valid values 0 to 15] |
| *rx_lvl* | : RX Trigger level [Valid values 0 to 15] |

**Returns**

Nothing

**Note**

When *tx_lvl* = 0; trigger will happen when TX FIFO is empty if *tx_lvl* = 15; trigger will happen when TX FIFO has at least one free space
When *rx_lvl* = 0; trigger will happen when RX FIFO has at least one byte in it, if *rx_lvl* = 15; trigger will happen when RX FIFO is full and cannot receive anymore data

Definition at line 452 of file uart_5411x.h.

#### 6.47.3.36 __STATIC_INLINE void Chip_UART_TXDisable ( LPC_USART_T ∗ *pUART* )

Disable transmission on UART TxD pin.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected pUART peripheral |

**Returns**

Nothing

Definition at line 274 of file uart_5411x.h.

#### 6.47.3.37 __STATIC_INLINE void Chip_UART_TXEnable ( LPC_USART_T ∗ *pUART* )

Enable transmission on UART TxD pin.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected pUART peripheral |

**Returns**

Nothing

Definition at line 264 of file uart_5411x.h.

#### 6.47.3.38 void Chip_UART_TXIntHandlerRB ( LPC_USART_T ∗ *pUART,* RINGBUFF_T ∗ *pRB* )

UART transmit-only interrupt handler for ring buffers.

**Parameters**

| | |
|---|---|
| *pUART* | : Pointer to selected UART peripheral |

| | |
|---|---|
| *pRB* | : Pointer to ring buffer structure to use |

**Returns**

Nothing

**Note**

If ring buffer support is desired for the transmit side of data transfer, the UART interrupt should call this function for a transmit based interrupt status.

Definition at line 289 of file uart_5411x.c.

## 6.48 CHIP: LPC5411X Windowed Watchdog driver

### 6.48.1 Detailed Description

### Data Structures

- struct LPC_WWDT_T

  *Windowed Watchdog register block structure.*

### Macros

- #define WWDT_WDMOD_BITMASK ((uint32_t) 0x3F)

  *Watchdog Mode register definitions.*
- #define WWDT_WDMOD_WDEN ((uint32_t) (1 << 0))
- #define WWDT_WDMOD_WDRESET ((uint32_t) (1 << 1))
- #define WWDT_WDMOD_WDTOF ((uint32_t) (1 << 2))
- #define WWDT_WDMOD_WDINT ((uint32_t) (1 << 3))
- #define WWDT_WDMOD_WDPROTECT ((uint32_t) (1 << 4))
- #define WWDT_WDMOD_LOCK ((uint32_t) (1 << 5))

### Functions

- __STATIC_INLINE void Chip_WWDT_Init (LPC_WWDT_T ∗pWWDT)

  *Initialize the Watchdog timer.*
- __STATIC_INLINE void Chip_WWDT_DeInit (LPC_WWDT_T ∗pWWDT)

  *Shutdown the Watchdog timer.*
- __STATIC_INLINE void Chip_WWDT_SetTimeOut (LPC_WWDT_T ∗pWWDT, uint32_t timeout)

  *Set WDT timeout constant value used for feed.*
- __STATIC_INLINE void Chip_WWDT_Feed (LPC_WWDT_T ∗pWWDT)

  *Feed watchdog timer.*
- __STATIC_INLINE void Chip_WWDT_SetWarning (LPC_WWDT_T ∗pWWDT, uint32_t timeout)

  *Set WWDT warning interrupt.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetWarning (LPC_WWDT_T ∗pWWDT)

  *Get WWDT warning interrupt.*
- __STATIC_INLINE void Chip_WWDT_SetWindow (LPC_WWDT_T ∗pWWDT, uint32_t timeout)

  *Set WWDT window time.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetWindow (LPC_WWDT_T ∗pWWDT)

  *Get WWDT window time.*
- __STATIC_INLINE void Chip_WWDT_SetOption (LPC_WWDT_T ∗pWWDT, uint32_t options)

  *Enable watchdog timer options.*
- __STATIC_INLINE void Chip_WWDT_UnsetOption (LPC_WWDT_T ∗pWWDT, uint32_t options)

  *Disable/clear watchdog timer options.*
- __STATIC_INLINE void Chip_WWDT_Start (LPC_WWDT_T ∗pWWDT)

  *Enable WWDT activity.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetStatus (LPC_WWDT_T ∗pWWDT)

  *Read WWDT status flag.*
- __STATIC_INLINE void Chip_WWDT_ClearStatusFlag (LPC_WWDT_T ∗pWWDT, uint32_t status)

  *Clear WWDT interrupt status flags.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetCurrentCount (LPC_WWDT_T ∗pWWDT)

  *Get the current value of WDT.*

## 6.48.2 Macro Definition Documentation

### 6.48.2.1 #define WWDT_WDMOD_BITMASK ((uint32_t) 0x3F)

Watchdog Mode register definitions.

Watchdog Mode Bitmask

Definition at line 61 of file wwdt_5411x.h.

### 6.48.2.2 #define WWDT_WDMOD_LOCK ((uint32_t) (1 << 5))

WWDT lock bit

Definition at line 73 of file wwdt_5411x.h.

### 6.48.2.3 #define WWDT_WDMOD_WDEN ((uint32_t) (1 << 0))

WWDT enable bit

Definition at line 63 of file wwdt_5411x.h.

### 6.48.2.4 #define WWDT_WDMOD_WDINT ((uint32_t) (1 << 3))

WWDT warning interrupt flag bit

Definition at line 69 of file wwdt_5411x.h.

### 6.48.2.5 #define WWDT_WDMOD_WDPROTECT ((uint32_t) (1 << 4))

WWDT Protect flag bit

Definition at line 71 of file wwdt_5411x.h.

### 6.48.2.6 #define WWDT_WDMOD_WDRESET ((uint32_t) (1 << 1))

WWDT reset enable bit

Definition at line 65 of file wwdt_5411x.h.

### 6.48.2.7 #define WWDT_WDMOD_WDTOF ((uint32_t) (1 << 2))

WWDT time-out flag bit

Definition at line 67 of file wwdt_5411x.h.

## 6.48.3 Function Documentation

### 6.48.3.1 __STATIC_INLINE void Chip_WWDT_ClearStatusFlag ( LPC_WWDT_T * *pWWDT,* uint32_t *status* )

Clear WWDT interrupt status flags.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |
| *status* | : Or'ed value of status flag(s) that you want to clear, should be:<br>• WWDT_WDMOD_WDTOF: Clear watchdog timeout flag<br>• WWDT_WDMOD_WDINT: Clear watchdog warning flag |

**Returns**

None

Definition at line 234 of file wwdt_5411x.h.

#### 6.48.3.2 __STATIC_INLINE void Chip_WWDT_DeInit ( LPC_WWDT_T ∗ *pWWDT* )

Shutdown the Watchdog timer.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

None

Definition at line 97 of file wwdt_5411x.h.

#### 6.48.3.3 __STATIC_INLINE void Chip_WWDT_Feed ( LPC_WWDT_T ∗ *pWWDT* )

Feed watchdog timer.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

None

**Note**

If this function isn't called, a watchdog timer warning will occur. After the warning, a timeout will occur if a feed has happened. Note that if WWDT registers are modified in an interrupt then it is a good idea to prevent those interrupts when writing the feed sequence.

Definition at line 122 of file wwdt_5411x.h.

#### 6.48.3.4 __STATIC_INLINE uint32_t Chip_WWDT_GetCurrentCount ( LPC_WWDT_T ∗ *pWWDT* )

Get the current value of WDT.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

current value of WDT

Definition at line 250 of file wwdt_5411x.h.

#### 6.48.3.5 __STATIC_INLINE uint32_t Chip_WWDT_GetStatus ( LPC_WWDT_T ∗ *pWWDT* )

Read WWDT status flag.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

Watchdog status, an Or'ed value of WWDT_WDMOD_∗

Definition at line 221 of file wwdt_5411x.h.

#### 6.48.3.6 __STATIC_INLINE uint32_t Chip_WWDT_GetWarning ( LPC_WWDT_T ∗ *pWWDT* )

Get WWDT warning interrupt.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

WWDT warning interrupt

Definition at line 146 of file wwdt_5411x.h.

#### 6.48.3.7 __STATIC_INLINE uint32_t Chip_WWDT_GetWindow ( LPC_WWDT_T ∗ *pWWDT* )

Get WWDT window time.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

WWDT window time

Definition at line 170 of file wwdt_5411x.h.

#### 6.48.3.8 __STATIC_INLINE void Chip_WWDT_Init ( LPC_WWDT_T ∗ *pWWDT* )

Initialize the Watchdog timer.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

None

Definition at line 80 of file wwdt_5411x.h.

#### 6.48.3.9 __STATIC_INLINE void Chip_WWDT_SetOption ( LPC_WWDT_T ∗ *pWWDT,* uint32_t *options* )

Enable watchdog timer options.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |
| *options* | : An or'ed set of options of values WWDT_WDMOD_WDEN, WWDT_WDMOD_WDRESET, and WWDT_WDMOD_WDPROTECT |

**Returns**

None

**Note**

You can enable more than one option at once (ie, WWDT_WDMOD_WDRESET | WWDT_WDMOD_WDPR←OTECT), but use the WWDT_WDMOD_WDEN after all other options are set (or unset) with no other options. If WWDT_WDMOD_LOCK is used, it cannot be unset.

Definition at line 186 of file wwdt_5411x.h.

#### 6.48.3.10 __STATIC_INLINE void Chip_WWDT_SetTimeOut ( LPC_WWDT_T ∗ *pWWDT,* uint32_t *timeout* )

Set WDT timeout constant value used for feed.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |
| *timeout* | : WDT timeout in ticks, between WWDT_TICKS_MIN and WWDT_TICKS_MAX |

**Returns**

none

Definition at line 108 of file wwdt_5411x.h.

#### 6.48.3.11 __STATIC_INLINE void Chip_WWDT_SetWarning ( LPC_WWDT_T ∗ *pWWDT,* uint32_t *timeout* )

Set WWDT warning interrupt.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

| | |
|---|---|
| *timeout* | : WDT warning in ticks, between 0 and 1023 |

**Returns**

None

**Note**

This is the number of ticks after the watchdog interrupt that the warning interrupt will be generated.

Definition at line 136 of file wwdt_5411x.h.

#### 6.48.3.12 __STATIC_INLINE void Chip_WWDT_SetWindow ( LPC_WWDT_T ∗ *pWWDT,* uint32_t *timeout* )

Set WWDT window time.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |
| *timeout* | : WDT timeout in ticks, between WWDT_TICKS_MIN and WWDT_TICKS_MAX |

**Returns**

None

**Note**

The watchdog timer must be fed between the timeout from the Chip_WWDT_SetTimeOut() function and this function, with this function defining the last tick before the watchdog window interrupt occurs.

Definition at line 160 of file wwdt_5411x.h.

#### 6.48.3.13 __STATIC_INLINE void Chip_WWDT_Start ( LPC_WWDT_T ∗ *pWWDT* )

Enable WWDT activity.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

**Returns**

None

Definition at line 210 of file wwdt_5411x.h.

#### 6.48.3.14 __STATIC_INLINE void Chip_WWDT_UnsetOption ( LPC_WWDT_T ∗ *pWWDT,* uint32_t *options* )

Disable/clear watchdog timer options.

**Parameters**

| | |
|---|---|
| *pWWDT* | : The base of WatchDog Timer peripheral on the chip |

|  |  |
|---|---|
| *options* | : An or'ed set of options of values WWDT_WDMOD_WDEN, WWDT_WDMOD_WDRESET, and WWDT_WDMOD_WDPROTECT |

**Returns**

None

**Note**

You can disable more than one option at once (ie, WWDT_WDMOD_WDRESET | WWDT_WDMOD_WDT↩ OF).

Definition at line 200 of file wwdt_5411x.h.

# 6.49 CHIP: LPC5411X flexcomm API

## 6.49.1 Detailed Description

### Macros

- #define FLEXCOMM_LOCK (1 << 3)

  *FlexCOMM PSEL register bits.*
- #define FLEXCOMM_ID_USART (1 << 4)
- #define FLEXCOMM_ID_SPI (1 << 5)
- #define FLEXCOMM_ID_I2C (1 << 6)
- #define FLEXCOMM_ID_I2S (1 << 7)
- #define ERR_FLEXCOMM_FUNCNOTSUPPORTED -1

  *Flexcomm error and offset values.*
- #define ERR_FLEXCOMM_NOTFREE -2
- #define ERR_FLEXCOMM_INVALIDBASE -3
- #define FLEXCOMM_PSEL_OFFSET 0xFF8

### Typedefs

- typedef void LPC_FLEXCOMM_T

### Enumerations

- enum FLEXCOMM_PERIPH_T {
  FLEXCOMM_PERIPH_NONE, FLEXCOMM_PERIPH_USART, FLEXCOMM_PERIPH_SPI, FLEXCOMM_PERIPH_I2C,
  FLEXCOMM_PERIPH_I2S_TX, FLEXCOMM_PERIPH_I2S_RX }

  *FLEXCOMM Peripheral functions.*

### Functions

- __STATIC_INLINE FLEXCOMM_PERIPH_T Chip_FLEXCOMM_GetFunc (LPC_FLEXCOMM_T ∗pFCOMM)

  *Get currently enabled FLEXCOMM function.*
- __STATIC_INLINE int Chip_FLEXCOMM_IsLocked (LPC_FLEXCOMM_T ∗pFCOMM)

  *Checks if given FLEXCOMM is locked to a function.*
- __STATIC_INLINE void Chip_FLEXCOMM_Lock (LPC_FLEXCOMM_T ∗pFCOMM)

  *Lock FLEXCOMM to a function.*
- int Chip_FLEXCOMM_SetPeriph (LPC_FLEXCOMM_T ∗pFCOMM, FLEXCOMM_PERIPH_T periph, int lock)

  *Set FLEXCOMM to a peripheral function.*
- int Chip_FLEXCOMM_GetIndex (LPC_FLEXCOMM_T ∗pFCOMM)

  *Get index of the FLEXCOMM corresponding to the given base address.*
- int Chip_FLEXCOMM_Init (LPC_FLEXCOMM_T ∗pFCOMM, FLEXCOMM_PERIPH_T periph)

  *Initialize FlexCOMM and associate it with a given peripheral.*
- void Chip_FLEXCOMM_DeInit (LPC_FLEXCOMM_T ∗pFCOMM)

  *Uninitialize the FlexCOMM.*

## 6.49.2 Macro Definition Documentation

### 6.49.2.1 #define ERR_FLEXCOMM_FUNCNOTSUPPORTED -1

Flexcomm error and offset values.

Selected flexcomm does not support the function desired

Definition at line 65 of file flexcomm_5411x.h.

### 6.49.2.2 #define ERR_FLEXCOMM_INVALIDBASE -3

Invalid flexcomm base address

Definition at line 67 of file flexcomm_5411x.h.

### 6.49.2.3 #define ERR_FLEXCOMM_NOTFREE -2

FlexCOMM is not free (used as a different peripheral)

Definition at line 66 of file flexcomm_5411x.h.

### 6.49.2.4 #define FLEXCOMM_ID_I2C (1 << 6)

I2C Support bit

Definition at line 61 of file flexcomm_5411x.h.

### 6.49.2.5 #define FLEXCOMM_ID_I2S (1 << 7)

I2S Support bit

Definition at line 62 of file flexcomm_5411x.h.

### 6.49.2.6 #define FLEXCOMM_ID_SPI (1 << 5)

SPI Support bit

Definition at line 60 of file flexcomm_5411x.h.

### 6.49.2.7 #define FLEXCOMM_ID_USART (1 << 4)

USART Support bit

Definition at line 59 of file flexcomm_5411x.h.

### 6.49.2.8 #define FLEXCOMM_LOCK (1 << 3)

FlexCOMM PSEL register bits.

Lock Bit

Definition at line 58 of file flexcomm_5411x.h.

### 6.49.2.9 #define FLEXCOMM_PSEL_OFFSET 0xFF8

Offset of the flexcomm PSEL register

Definition at line 68 of file flexcomm_5411x.h.

## 6.49.3 Typedef Documentation

### 6.49.3.1 typedef void **LPC_FLEXCOMM_T**

Definition at line 45 of file flexcomm_5411x.h.

## 6.49.4 Enumeration Type Documentation

### 6.49.4.1 enum **FLEXCOMM_PERIPH_T**

FLEXCOMM Peripheral functions.

**Enumerator**

***FLEXCOMM_PERIPH_NONE*** No peripheral

***FLEXCOMM_PERIPH_USART*** USART peripheral

***FLEXCOMM_PERIPH_SPI*** SPI Peripheral

***FLEXCOMM_PERIPH_I2C*** I2C Peripheral

***FLEXCOMM_PERIPH_I2S_TX*** I2S TX Peripheral

***FLEXCOMM_PERIPH_I2S_RX*** I2S RX Peripheral

Definition at line 48 of file flexcomm_5411x.h.

## 6.49.5 Function Documentation

### 6.49.5.1 void Chip_FLEXCOMM_DeInit ( **LPC_FLEXCOMM_T** ∗ *pFCOMM* )

Uninitialize the FlexCOMM.

**Parameters**

| | |
|---|---|
| *pFCOMM* | : Base address of the flexcomm peripheral |

**Returns**

Nothing

Definition at line 106 of file flexcomm_5411x.c.

### 6.49.5.2 __STATIC_INLINE **FLEXCOMM_PERIPH_T** Chip_FLEXCOMM_GetFunc ( **LPC_FLEXCOMM_T** ∗ *pFCOMM* )

Get currently enabled FLEXCOMM function.

**Parameters**

| | |
|---|---|
| *pFCOMM* | : Base address of the flexcomm peripheral |

**Returns**

Enabled flexcomm peripheral (See FLEXCOMM_PERIPH_T)

Definition at line 75 of file flexcomm_5411x.h.

#### 6.49.5.3 int Chip_FLEXCOMM_GetIndex ( LPC_FLEXCOMM_T ∗ *pFCOMM* )

Get index of the FLEXCOMM corresponding to the given base address.

**Parameters**

| | |
|---|---|
| *pFCOMM* | : Base address of the flexcomm peripheral |

**Returns**

Index of FLEXCOMM, ERR_FLEXCOMM_INVALIDBASE - when base address is invalid

Definition at line 51 of file flexcomm_5411x.c.

#### 6.49.5.4 int Chip_FLEXCOMM_Init ( LPC_FLEXCOMM_T ∗ *pFCOMM,* FLEXCOMM_PERIPH_T *periph* )

Initialize FlexCOMM and associate it with a given peripheral.

**Parameters**

| | |
|---|---|
| *pFCOMM* | : Base address of the flexcomm peripheral |
| *periph* | : Peripheral to set the FlexCOMM to |

**Returns**

0 on success, ERR_FLEXCOMM_FUNCNOTSUPPORTED when the given FlexComm does not support peripheral provided by *periph*, ERR_FLEXCOMM_NOTFREE when the flexcomm is being used already as a peripheral.

Definition at line 76 of file flexcomm_5411x.c.

#### 6.49.5.5 __STATIC_INLINE int Chip_FLEXCOMM_IsLocked ( LPC_FLEXCOMM_T ∗ *pFCOMM* )

Checks if given FLEXCOMM is locked to a function.

**Parameters**

| | |
|---|---|
| *pFCOMM* | : Base address of the flexcomm peripheral |

**Returns**

1 - FLEXCOMM is locked to a function; 0 - FLEXCOMM is not locked

Definition at line 85 of file flexcomm_5411x.h.

#### 6.49.5.6 __STATIC_INLINE void Chip_FLEXCOMM_Lock ( LPC_FLEXCOMM_T ∗ *pFCOMM* )

Lock FLEXCOMM to a function.

**Parameters**

| | |
|---|---|
| *pFCOMM* | : Base address of the flexcomm peripheral |

**Returns**

Nothing

**Note**

Once the FLEXCOMM is locked, it can only be unlocked by a reset

Definition at line 96 of file flexcomm_5411x.h.

### 6.49.5.7 int Chip_FLEXCOMM_SetPeriph ( LPC_FLEXCOMM_T ∗ *pFCOMM*, FLEXCOMM_PERIPH_T *periph*, int *lock* )

Set FLEXCOMM to a peripheral function.

#### Parameters

| | |
|---|---|
| *pFCOMM* | : Base address of the flexcomm peripheral |
| *periph* | : Selected peripheral (See FLEXCOMM_PERIPH_T) |
| *lock* | : 1 - Flexcomm will be locked as given peripheral, 0 - Do not lock |

#### See also

Chip_FLEXCOMM_Init()

#### Returns

0 on success, ERR_FLEXCOMM_FUNCNOTSUPPORTED when the given FlexComm does not support peripheral provided by *periph*, ERR_FLEXCOMM_NOTFREE when the flexcomm is being used already as a peripheral.

#### Note

Once the FLEXCOMM is locked, it can only be unlocked by a reset

Definition at line 128 of file flexcomm_5411x.c.

# 6.50 CHIP: LPC5411X support functions

## 6.50.1 Detailed Description

### Functions

- void SystemCoreClockUpdate (void)

  *Update system core and ASYNC syscon clock rate, should be called if the system has a clock rate change.*
- void Chip_SystemInit (void)

  *Set up and initialize hardware prior to call to main()*
- void Chip_SetupIrcClocking (uint32_t iFreq)

  *Clock and PLL initialization based on the internal oscillator.*
- void Chip_SetupExtInClocking (uint32_t iFreq)

  *Clock and PLL initialization based on the external clock input.*
- void Chip_SetupFROClocking (uint32_t iFreq)

  *Initialize the Core clock to given frequency (12, 48 or 96 MHz)*
- void Chip_USB_Init (void)

  *Initialize the USB bus.*
- __STATIC_INLINE void Chip_USB_TrimOff (int enable)

  *Turn of FRO clock trimming based on USB SOF.*

### Variables

- uint32_t SystemCoreClock

  *Current system clock rate, mainly used for peripherals in SYSCON.*

## 6.50.2 Function Documentation

### 6.50.2.1 void Chip_SetupExtInClocking ( uint32_t *iFreq* )

Clock and PLL initialization based on the external clock input.

**Parameters**

| | |
|---|---|
| *iFreq* | : Rate (in Hz) to set the main system clock to |

**Returns**

None

Definition at line 141 of file sysinit_5411x.c.

### 6.50.2.2 void Chip_SetupFROClocking ( uint32_t *iFreq* )

Initialize the Core clock to given frequency (12, 48 or 96 MHz)

**Parameters**

| | |
|---|---|
| *iFreq* | : Desired frequency (must be one of SYSCON_FRO12MHZ_FREQ or SYSCON_FRO48MHZ_FREQ or SYSCON_FRO96MHZ_FREQ) |

**Returns**

Nothing

Definition at line 70 of file sysinit_5411x.c.

#### 6.50.2.3 void Chip_SetupIrcClocking ( uint32_t *iFreq* )

Clock and PLL initialization based on the internal oscillator.

**Parameters**

| | |
|---|---|
| *iFreq* | : Rate (in Hz) to set the main system clock to |

**Returns**

None

Definition at line 97 of file sysinit_5411x.c.

#### 6.50.2.4 void Chip_SystemInit ( void )

Set up and initialize hardware prior to call to main()

**Returns**

None

**Note**

Chip_SystemInit() is called prior to the application and sets up system clocking prior to the application starting.

Definition at line 191 of file sysinit_5411x.c.

#### 6.50.2.5 void Chip_USB_Init ( void )

Initialize the USB bus.

**Returns**

Nothing

**Note**

Uses FRO HF to initialize the pin-IO and the clocks.

Definition at line 61 of file chip_5411x.c.

#### 6.50.2.6 __STATIC_INLINE void Chip_USB_TrimOff ( int *enable* )

Turn of FRO clock trimming based on USB SOF.

**Parameters**

| | |
|---|---|
| *enable* | : 0 - Disable trim based on USB SOF; Non-Zero to enable it |

**Returns**

Nothing

Definition at line 271 of file chip.h.

#### 6.50.2.7 void SystemCoreClockUpdate ( void )

Update system core and ASYNC syscon clock rate, should be called if the system has a clock rate change.

**Returns**

None

Definition at line 55 of file chip_5411x.c.

### 6.50.3 Variable Documentation

#### 6.50.3.1 uint32_t SystemCoreClock

Current system clock rate, mainly used for peripherals in SYSCON.

Definition at line 43 of file chip_5411x.c.

## 6.51 CHIP: LPC5411x Chip driver build time options

### 6.51.1 Detailed Description

Some chip drivers require build-time configuration to enable and disable specific platform features. A build-time option is configured by the use of a definition passed to the compiler during the build process or by adding the definition to the sys_config.h file.

**CORE_M4 definition**
LPC5411x devices should add a unvalued CORE_M4 definition to the compilers argument list when building images for the M4 core. Do not add this definition as part of sys_config.h.

**CORE_M0PLUS definition**
LPC5411x devices should add a unvalued CORE_M0PLUS definition to the compilers argument list when building images for the M0+ core. Do not add this definition as part of sys_config.h.

**CHIP_LPC5411X definition**
CHIP_LPC5411X must be defined for all code when building with Keil and IAR for the LPC5411x device. This unvalued definition can be added to the compilers argument list or as part of sys_config.h.

### 6.51.2 Variable Documentation

#### 6.51.2.1 const uint32_t ExtClockIn

Clock rate on the CLKIN pin This value is defined externally to the chip layer and contains the value in Hz for the CLKIN pin for the board. If this pin isn't used, this rate can be 0.

# 6.52 CHIP: LPC5411x I2C driver

## 6.52.1 Detailed Description

### Modules

- CHIP: LPC5411X I2C master-only driver
- CHIP: LPC5411X I2C slave-only driver

### Macros

- #define I2C_CFG_MSTEN (1 << 0)
- #define I2C_CFG_MSTEN (1 << 0)
- #define I2C_CFG_SLVEN (1 << 1)
- #define I2C_CFG_SLVEN (1 << 1)
- #define I2C_CFG_MONEN (1 << 2)
- #define I2C_CFG_MONEN (1 << 2)
- #define I2C_CFG_TIMEOUTEN (1 << 3)
- #define I2C_CFG_TIMEOUTEN (1 << 3)
- #define I2C_CFG_MONCLKSTR (1 << 4)
- #define I2C_CFG_MONCLKSTR (1 << 4)
- #define I2C_CFG_MASK ((uint32_t) 0x1F)
- #define I2C_CFG_MASK ((uint32_t) 0x1F)
- #define I2C_STAT_MSTPENDING (1 << 0)
- #define I2C_STAT_MSTPENDING (1 << 0)
- #define I2C_STAT_MSTSTATE (0x7 << 1)
- #define I2C_STAT_MSTSTATE (0x7 << 1)
- #define I2C_STAT_MSTRARBLOSS (1 << 4)
- #define I2C_STAT_MSTRARBLOSS (1 << 4)
- #define I2C_STAT_MSTSTSTPERR (1 << 6)
- #define I2C_STAT_MSTSTSTPERR (1 << 6)
- #define I2C_STAT_SLVPENDING (1 << 8)
- #define I2C_STAT_SLVPENDING (1 << 8)
- #define I2C_STAT_SLVSTATE (0x3 << 9)
- #define I2C_STAT_SLVSTATE (0x3 << 9)
- #define I2C_STAT_SLVNOTSTR (1 << 11)
- #define I2C_STAT_SLVNOTSTR (1 << 11)
- #define I2C_STAT_SLVIDX (0x3 << 12)
- #define I2C_STAT_SLVIDX (0x3 << 12)
- #define I2C_STAT_SLVSEL (1 << 14)
- #define I2C_STAT_SLVSEL (1 << 14)
- #define I2C_STAT_SLVDESEL (1 << 15)
- #define I2C_STAT_SLVDESEL (1 << 15)
- #define I2C_STAT_MONRDY (1 << 16)
- #define I2C_STAT_MONRDY (1 << 16)
- #define I2C_STAT_MONOV (1 << 17)
- #define I2C_STAT_MONOV (1 << 17)
- #define I2C_STAT_MONACTIVE (1 << 18)
- #define I2C_STAT_MONACTIVE (1 << 18)
- #define I2C_STAT_MONIDLE (1 << 19)
- #define I2C_STAT_MONIDLE (1 << 19)
- #define I2C_STAT_EVENTTIMEOUT (1 << 24)
- #define I2C_STAT_EVENTTIMEOUT (1 << 24)
- #define I2C_STAT_SCLTIMEOUT (1 << 25)

- #define I2C_STAT_SCLTIMEOUT (1 << 25)
- #define I2C_STAT_MSTCODE_IDLE (0)
- #define I2C_STAT_MSTCODE_IDLE (0)
- #define I2C_STAT_MSTCODE_RXREADY (1)
- #define I2C_STAT_MSTCODE_RXREADY (1)
- #define I2C_STAT_MSTCODE_TXREADY (2)
- #define I2C_STAT_MSTCODE_TXREADY (2)
- #define I2C_STAT_MSTCODE_NACKADR (3)
- #define I2C_STAT_MSTCODE_NACKADR (3)
- #define I2C_STAT_MSTCODE_NACKDAT (4)
- #define I2C_STAT_MSTCODE_NACKDAT (4)
- #define I2C_STAT_SLVCODE_ADDR (0)
- #define I2C_STAT_SLVCODE_ADDR (0)
- #define I2C_STAT_SLVCODE_RX (1)
- #define I2C_STAT_SLVCODE_RX (1)
- #define I2C_STAT_SLVCODE_TX (2)
- #define I2C_STAT_SLVCODE_TX (2)
- #define I2C_INTENSET_MSTPENDING (1 << 0)
- #define I2C_INTENSET_MSTPENDING (1 << 0)
- #define I2C_INTENSET_MSTRARBLOSS (1 << 4)
- #define I2C_INTENSET_MSTRARBLOSS (1 << 4)
- #define I2C_INTENSET_MSTSTSTPERR (1 << 6)
- #define I2C_INTENSET_MSTSTSTPERR (1 << 6)
- #define I2C_INTENSET_SLVPENDING (1 << 8)
- #define I2C_INTENSET_SLVPENDING (1 << 8)
- #define I2C_INTENSET_SLVNOTSTR (1 << 11)
- #define I2C_INTENSET_SLVNOTSTR (1 << 11)
- #define I2C_INTENSET_SLVDESEL (1 << 15)
- #define I2C_INTENSET_SLVDESEL (1 << 15)
- #define I2C_INTENSET_MONRDY (1 << 16)
- #define I2C_INTENSET_MONRDY (1 << 16)
- #define I2C_INTENSET_MONOV (1 << 17)
- #define I2C_INTENSET_MONOV (1 << 17)
- #define I2C_INTENSET_MONIDLE (1 << 19)
- #define I2C_INTENSET_MONIDLE (1 << 19)
- #define I2C_INTENSET_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENSET_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENSET_SCLTIMEOUT (1 << 25)
- #define I2C_INTENSET_SCLTIMEOUT (1 << 25)
- #define I2C_INTENCLR_MSTPENDING (1 << 0)
- #define I2C_INTENCLR_MSTPENDING (1 << 0)
- #define I2C_INTENCLR_MSTRARBLOSS (1 << 4)
- #define I2C_INTENCLR_MSTRARBLOSS (1 << 4)
- #define I2C_INTENCLR_MSTSTSTPERR (1 << 6)
- #define I2C_INTENCLR_MSTSTSTPERR (1 << 6)
- #define I2C_INTENCLR_SLVPENDING (1 << 8)
- #define I2C_INTENCLR_SLVPENDING (1 << 8)
- #define I2C_INTENCLR_SLVNOTSTR (1 << 11)
- #define I2C_INTENCLR_SLVNOTSTR (1 << 11)
- #define I2C_INTENCLR_SLVDESEL (1 << 15)
- #define I2C_INTENCLR_SLVDESEL (1 << 15)
- #define I2C_INTENCLR_MONRDY (1 << 16)
- #define I2C_INTENCLR_MONRDY (1 << 16)
- #define I2C_INTENCLR_MONOV (1 << 17)
- #define I2C_INTENCLR_MONOV (1 << 17)

- #define I2C_INTENCLR_MONIDLE (1 << 19)
- #define I2C_INTENCLR_MONIDLE (1 << 19)
- #define I2C_INTENCLR_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENCLR_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENCLR_SCLTIMEOUT (1 << 25)
- #define I2C_INTENCLR_SCLTIMEOUT (1 << 25)
- #define I2C_TIMEOUT_VAL(n) (((uint32_t) ((n) - 1) & 0xFFF0) | 0x000F)
- #define I2C_TIMEOUT_VAL(n) (((uint32_t) ((n) - 1) & 0xFFF0) | 0x000F)
- #define I2C_INTSTAT_MSTPENDING (1 << 0)
- #define I2C_INTSTAT_MSTPENDING (1 << 0)
- #define I2C_INTSTAT_MSTRARBLOSS (1 << 4)
- #define I2C_INTSTAT_MSTRARBLOSS (1 << 4)
- #define I2C_INTSTAT_MSTSTSTPERR (1 << 6)
- #define I2C_INTSTAT_MSTSTSTPERR (1 << 6)
- #define I2C_INTSTAT_SLVPENDING (1 << 8)
- #define I2C_INTSTAT_SLVPENDING (1 << 8)
- #define I2C_INTSTAT_SLVNOTSTR (1 << 11)
- #define I2C_INTSTAT_SLVNOTSTR (1 << 11)
- #define I2C_INTSTAT_SLVDESEL (1 << 15)
- #define I2C_INTSTAT_SLVDESEL (1 << 15)
- #define I2C_INTSTAT_MONRDY (1 << 16)
- #define I2C_INTSTAT_MONRDY (1 << 16)
- #define I2C_INTSTAT_MONOV (1 << 17)
- #define I2C_INTSTAT_MONOV (1 << 17)
- #define I2C_INTSTAT_MONIDLE (1 << 19)
- #define I2C_INTSTAT_MONIDLE (1 << 19)
- #define I2C_INTSTAT_EVENTTIMEOUT (1 << 24)
- #define I2C_INTSTAT_EVENTTIMEOUT (1 << 24)
- #define I2C_INTSTAT_SCLTIMEOUT (1 << 25)
- #define I2C_INTSTAT_SCLTIMEOUT (1 << 25)
- #define I2C_MSTCTL_MSTCONTINUE (1 << 0)
- #define I2C_MSTCTL_MSTCONTINUE (1 << 0)
- #define I2C_MSTCTL_MSTSTART (1 << 1)
- #define I2C_MSTCTL_MSTSTART (1 << 1)
- #define I2C_MSTCTL_MSTSTOP (1 << 2)
- #define I2C_MSTCTL_MSTSTOP (1 << 2)
- #define I2C_MSTCTL_MSTDMA (1 << 3)
- #define I2C_MSTCTL_MSTDMA (1 << 3)
- #define I2C_MSTTIME_MSTSCLLOW (0x07 << 0)
- #define I2C_MSTTIME_MSTSCLLOW (0x07 << 0)
- #define I2C_MSTTIME_MSTSCLHIGH (0x07 << 4)
- #define I2C_MSTTIME_MSTSCLHIGH (0x07 << 4)
- #define I2C_MSTDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_MSTDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_SLVCTL_SLVCONTINUE (1 << 0)
- #define I2C_SLVCTL_SLVCONTINUE (1 << 0)
- #define I2C_SLVCTL_SLVNACK (1 << 1)
- #define I2C_SLVCTL_SLVNACK (1 << 1)
- #define I2C_SLVCTL_SLVDMA (1 << 3)
- #define I2C_SLVCTL_SLVDMA (1 << 3)
- #define I2C_SLVDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_SLVDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_SLVADR_SADISABLE (1 << 0)
- #define I2C_SLVADR_SADISABLE (1 << 0)
- #define I2C_SLVADR_SLVADR (0x7F << 1)

- #define I2C_SLVADR_SLVADR (0x7F << 1)
- #define I2C_SLVADR_MASK ((uint32_t) 0x00FF)
- #define I2C_SLVADR_MASK ((uint32_t) 0x00FF)
- #define I2C_SLVQUAL_QUALMODE0 (1 << 0)
- #define I2C_SLVQUAL_QUALMODE0 (1 << 0)
- #define I2C_SLVQUAL_SLVQUAL0 (0x7F << 1)
- #define I2C_SLVQUAL_SLVQUAL0 (0x7F << 1)
- #define I2C_MONRXDAT_DATA (0xFF << 0)
- #define I2C_MONRXDAT_DATA (0xFF << 0)
- #define I2C_MONRXDAT_MONSTART (1 << 8)
- #define I2C_MONRXDAT_MONSTART (1 << 8)
- #define I2C_MONRXDAT_MONRESTART (1 << 9)
- #define I2C_MONRXDAT_MONRESTART (1 << 9)
- #define I2C_MONRXDAT_MONNACK (1 << 10)
- #define I2C_MONRXDAT_MONNACK (1 << 10)

## Functions

- __STATIC_INLINE int Chip_I2C_Init (LPC_I2C_T ∗pI2C)

  *Initialize I2C Interface.*
- __STATIC_INLINE void Chip_I2C_DeInit (LPC_I2C_T ∗pI2C)

  *Shutdown I2C Interface.*
- __STATIC_INLINE void Chip_I2C_SetClockDiv (LPC_I2C_T ∗pI2C, uint32_t clkdiv)

  *Sets I2C Clock Divider registers.*
- static INLINE uint32_t Chip_I2C_GetClockDiv (LPC_I2C_T ∗pI2C)

  *Get I2C Clock Divider registers.*
- static INLINE void Chip_I2C_EnableInt (LPC_I2C_T ∗pI2C, uint32_t intEn)

  *Enable I2C Interrupts.*
- static INLINE void Chip_I2C_DisableInt (LPC_I2C_T ∗pI2C, uint32_t intClr)

  *Disable I2C Interrupts.*
- static INLINE void Chip_I2C_ClearInt (LPC_I2C_T ∗pI2C, uint32_t intClr)

  *Disable I2C Interrupts.*
- static INLINE uint32_t Chip_I2C_GetPendingInt (LPC_I2C_T ∗pI2C)

  *Returns pending I2C Interrupts.*

## Variables

- __IO uint32_t STAT

  *I2C register block structure.*
- __IO uint32_t INTENSET
- __O uint32_t INTENCLR
- __IO uint32_t TIMEOUT
- __IO uint32_t CLKDIV
- __IO uint32_t INTSTAT
- __I uint32_t RESERVED0
- __IO uint32_t MSTCTL
- __IO uint32_t MSTTIME
- __IO uint32_t MSTDAT
- __IO uint32_t RESERVED1 [5]
- __IO uint32_t SLVCTL
- __IO uint32_t SLVDAT
- __IO uint32_t SLVADR [4]

- __IO uint32_t SLVQUAL0
- __IO uint32_t RESERVED2 [9]
- __I uint32_t MONRXDAT
- __IO uint32_t PSELID
- __I uint32_t PID
- LPC_I2C_T

## 6.52.2 Macro Definition Documentation

### 6.52.2.1 #define I2C_CFG_MASK ((uint32_t) 0x1F)

Configuration Register Mask

Definition at line 225 of file i2c_common_5411x.h.

### 6.52.2.2 #define I2C_CFG_MASK ((uint32_t) 0x1F)

Configuration Register Mask

Definition at line 225 of file i2c_common_5411x.h.

### 6.52.2.3 #define I2C_CFG_MONCLKSTR (1 << 4)

Monitor Clock Stretching Bit

Definition at line 224 of file i2c_common_5411x.h.

### 6.52.2.4 #define I2C_CFG_MONCLKSTR (1 << 4)

Monitor Clock Stretching Bit

Definition at line 224 of file i2c_common_5411x.h.

### 6.52.2.5 #define I2C_CFG_MONEN (1 << 2)

Monitor Enable/Disable Bit

Definition at line 222 of file i2c_common_5411x.h.

### 6.52.2.6 #define I2C_CFG_MONEN (1 << 2)

Monitor Enable/Disable Bit

Definition at line 222 of file i2c_common_5411x.h.

### 6.52.2.7 #define I2C_CFG_MSTEN (1 << 0)

Master Enable/Disable Bit

Definition at line 220 of file i2c_common_5411x.h.

### 6.52.2.8 #define I2C_CFG_MSTEN (1 << 0)

Master Enable/Disable Bit

Definition at line 220 of file i2c_common_5411x.h.

#### 6.52.2.9 #define I2C_CFG_SLVEN (1 << 1)

Slave Enable/Disable Bit

Definition at line 221 of file i2c_common_5411x.h.

#### 6.52.2.10 #define I2C_CFG_SLVEN (1 << 1)

Slave Enable/Disable Bit

Definition at line 221 of file i2c_common_5411x.h.

#### 6.52.2.11 #define I2C_CFG_TIMEOUTEN (1 << 3)

Timeout Enable/Disable Bit

Definition at line 223 of file i2c_common_5411x.h.

#### 6.52.2.12 #define I2C_CFG_TIMEOUTEN (1 << 3)

Timeout Enable/Disable Bit

Definition at line 223 of file i2c_common_5411x.h.

#### 6.52.2.13 #define I2C_INTENCLR_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Clear Bit

Definition at line 284 of file i2c_common_5411x.h.

#### 6.52.2.14 #define I2C_INTENCLR_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Clear Bit

Definition at line 284 of file i2c_common_5411x.h.

#### 6.52.2.15 #define I2C_INTENCLR_MONIDLE (1 << 19)

Monitor Idle Interrupt Clear Bit

Definition at line 283 of file i2c_common_5411x.h.

#### 6.52.2.16 #define I2C_INTENCLR_MONIDLE (1 << 19)

Monitor Idle Interrupt Clear Bit

Definition at line 283 of file i2c_common_5411x.h.

#### 6.52.2.17 #define I2C_INTENCLR_MONOV (1 << 17)

Monitor Overflow Interrupt Clear Bit

Definition at line 282 of file i2c_common_5411x.h.

#### 6.52.2.18 #define I2C_INTENCLR_MONOV (1 << 17)

Monitor Overflow Interrupt Clear Bit

Definition at line 282 of file i2c_common_5411x.h.

#### 6.52.2.19 #define I2C_INTENCLR_MONRDY (1 << 16)

Monitor Ready Interrupt Clear Bit

Definition at line 281 of file i2c_common_5411x.h.

#### 6.52.2.20 #define I2C_INTENCLR_MONRDY (1 << 16)

Monitor Ready Interrupt Clear Bit

Definition at line 281 of file i2c_common_5411x.h.

#### 6.52.2.21 #define I2C_INTENCLR_MSTPENDING (1 << 0)

Master Pending Interrupt Clear Bit

Definition at line 275 of file i2c_common_5411x.h.

#### 6.52.2.22 #define I2C_INTENCLR_MSTPENDING (1 << 0)

Master Pending Interrupt Clear Bit

Definition at line 275 of file i2c_common_5411x.h.

#### 6.52.2.23 #define I2C_INTENCLR_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Interrupt Clear Bit

Definition at line 276 of file i2c_common_5411x.h.

#### 6.52.2.24 #define I2C_INTENCLR_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Interrupt Clear Bit

Definition at line 276 of file i2c_common_5411x.h.

#### 6.52.2.25 #define I2C_INTENCLR_MSTSTSTPERR (1 << 6)

Master Start Stop Error Interrupt Clear Bit

Definition at line 277 of file i2c_common_5411x.h.

#### 6.52.2.26 #define I2C_INTENCLR_MSTSTSTPERR (1 << 6)

Master Start Stop Error Interrupt Clear Bit

Definition at line 277 of file i2c_common_5411x.h.

#### 6.52.2.27 #define I2C_INTENCLR_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Clear Bit

Definition at line 285 of file i2c_common_5411x.h.

#### 6.52.2.28 #define I2C_INTENCLR_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Clear Bit

Definition at line 285 of file i2c_common_5411x.h.

#### 6.52.2.29 #define I2C_INTENCLR_SLVDESEL (1 << 15)

Slave Deselect Interrupt Clear Bit

Definition at line 280 of file i2c_common_5411x.h.

#### 6.52.2.30 #define I2C_INTENCLR_SLVDESEL (1 << 15)

Slave Deselect Interrupt Clear Bit

Definition at line 280 of file i2c_common_5411x.h.

#### 6.52.2.31 #define I2C_INTENCLR_SLVNOTSTR (1 << 11)

Slave not stretching Clock Interrupt Clear Bit

Definition at line 279 of file i2c_common_5411x.h.

#### 6.52.2.32 #define I2C_INTENCLR_SLVNOTSTR (1 << 11)

Slave not stretching Clock Interrupt Clear Bit

Definition at line 279 of file i2c_common_5411x.h.

#### 6.52.2.33 #define I2C_INTENCLR_SLVPENDING (1 << 8)

Slave Pending Interrupt Clear Bit

Definition at line 278 of file i2c_common_5411x.h.

#### 6.52.2.34 #define I2C_INTENCLR_SLVPENDING (1 << 8)

Slave Pending Interrupt Clear Bit

Definition at line 278 of file i2c_common_5411x.h.

#### 6.52.2.35 #define I2C_INTENSET_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Enable Bit

Definition at line 269 of file i2c_common_5411x.h.

### 6.52.2.36 #define I2C_INTENSET_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Enable Bit

Definition at line 269 of file i2c_common_5411x.h.

### 6.52.2.37 #define I2C_INTENSET_MONIDLE (1 << 19)

Monitor Idle Interrupt Enable Bit

Definition at line 268 of file i2c_common_5411x.h.

### 6.52.2.38 #define I2C_INTENSET_MONIDLE (1 << 19)

Monitor Idle Interrupt Enable Bit

Definition at line 268 of file i2c_common_5411x.h.

### 6.52.2.39 #define I2C_INTENSET_MONOV (1 << 17)

Monitor Overflow Interrupt Enable Bit

Definition at line 267 of file i2c_common_5411x.h.

### 6.52.2.40 #define I2C_INTENSET_MONOV (1 << 17)

Monitor Overflow Interrupt Enable Bit

Definition at line 267 of file i2c_common_5411x.h.

### 6.52.2.41 #define I2C_INTENSET_MONRDY (1 << 16)

Monitor Ready Interrupt Enable Bit

Definition at line 266 of file i2c_common_5411x.h.

### 6.52.2.42 #define I2C_INTENSET_MONRDY (1 << 16)

Monitor Ready Interrupt Enable Bit

Definition at line 266 of file i2c_common_5411x.h.

### 6.52.2.43 #define I2C_INTENSET_MSTPENDING (1 << 0)

Master Pending Interrupt Enable Bit

Definition at line 260 of file i2c_common_5411x.h.

### 6.52.2.44 #define I2C_INTENSET_MSTPENDING (1 << 0)

Master Pending Interrupt Enable Bit

Definition at line 260 of file i2c_common_5411x.h.

#### 6.52.2.45 #define I2C_INTENSET_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Interrupt Enable Bit

Definition at line 261 of file i2c_common_5411x.h.

#### 6.52.2.46 #define I2C_INTENSET_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Interrupt Enable Bit

Definition at line 261 of file i2c_common_5411x.h.

#### 6.52.2.47 #define I2C_INTENSET_MSTSTSTPERR (1 << 6)

Master Start Stop Error Interrupt Enable Bit

Definition at line 262 of file i2c_common_5411x.h.

#### 6.52.2.48 #define I2C_INTENSET_MSTSTSTPERR (1 << 6)

Master Start Stop Error Interrupt Enable Bit

Definition at line 262 of file i2c_common_5411x.h.

#### 6.52.2.49 #define I2C_INTENSET_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Enable Bit

Definition at line 270 of file i2c_common_5411x.h.

#### 6.52.2.50 #define I2C_INTENSET_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Enable Bit

Definition at line 270 of file i2c_common_5411x.h.

#### 6.52.2.51 #define I2C_INTENSET_SLVDESEL (1 << 15)

Slave Deselect Interrupt Enable Bit

Definition at line 265 of file i2c_common_5411x.h.

#### 6.52.2.52 #define I2C_INTENSET_SLVDESEL (1 << 15)

Slave Deselect Interrupt Enable Bit

Definition at line 265 of file i2c_common_5411x.h.

#### 6.52.2.53 #define I2C_INTENSET_SLVNOTSTR (1 << 11)

Slave not stretching Clock Interrupt Enable Bit

Definition at line 264 of file i2c_common_5411x.h.

#### 6.52.2.54 #define I2C_INTENSET_SLVNOTSTR (1 << 11)

Slave not stretching Clock Interrupt Enable Bit

Definition at line 264 of file i2c_common_5411x.h.

#### 6.52.2.55 #define I2C_INTENSET_SLVPENDING (1 << 8)

Slave Pending Interrupt Enable Bit

Definition at line 263 of file i2c_common_5411x.h.

#### 6.52.2.56 #define I2C_INTENSET_SLVPENDING (1 << 8)

Slave Pending Interrupt Enable Bit

Definition at line 263 of file i2c_common_5411x.h.

#### 6.52.2.57 #define I2C_INTSTAT_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Status Bit

Definition at line 304 of file i2c_common_5411x.h.

#### 6.52.2.58 #define I2C_INTSTAT_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Status Bit

Definition at line 304 of file i2c_common_5411x.h.

#### 6.52.2.59 #define I2C_INTSTAT_MONIDLE (1 << 19)

Monitor Idle Interrupt Status Bit

Definition at line 303 of file i2c_common_5411x.h.

#### 6.52.2.60 #define I2C_INTSTAT_MONIDLE (1 << 19)

Monitor Idle Interrupt Status Bit

Definition at line 303 of file i2c_common_5411x.h.

#### 6.52.2.61 #define I2C_INTSTAT_MONOV (1 << 17)

Monitor Overflow Interrupt Status Bit

Definition at line 302 of file i2c_common_5411x.h.

#### 6.52.2.62 #define I2C_INTSTAT_MONOV (1 << 17)

Monitor Overflow Interrupt Status Bit

Definition at line 302 of file i2c_common_5411x.h.

#### 6.52.2.63 #define I2C_INTSTAT_MONRDY (1 << 16)

Monitor Ready Interrupt Status Bit

Definition at line 301 of file i2c_common_5411x.h.

#### 6.52.2.64 #define I2C_INTSTAT_MONRDY (1 << 16)

Monitor Ready Interrupt Status Bit

Definition at line 301 of file i2c_common_5411x.h.

#### 6.52.2.65 #define I2C_INTSTAT_MSTPENDING (1 << 0)

Master Pending Interrupt Status Bit

Definition at line 295 of file i2c_common_5411x.h.

#### 6.52.2.66 #define I2C_INTSTAT_MSTPENDING (1 << 0)

Master Pending Interrupt Status Bit

Definition at line 295 of file i2c_common_5411x.h.

#### 6.52.2.67 #define I2C_INTSTAT_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Interrupt Status Bit

Definition at line 296 of file i2c_common_5411x.h.

#### 6.52.2.68 #define I2C_INTSTAT_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Interrupt Status Bit

Definition at line 296 of file i2c_common_5411x.h.

#### 6.52.2.69 #define I2C_INTSTAT_MSTSTSTPERR (1 << 6)

Master Start Stop Error Interrupt Status Bit

Definition at line 297 of file i2c_common_5411x.h.

#### 6.52.2.70 #define I2C_INTSTAT_MSTSTSTPERR (1 << 6)

Master Start Stop Error Interrupt Status Bit

Definition at line 297 of file i2c_common_5411x.h.

#### 6.52.2.71 #define I2C_INTSTAT_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Status Bit

Definition at line 305 of file i2c_common_5411x.h.

#### 6.52.2.72 #define I2C_INTSTAT_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Status Bit

Definition at line 305 of file i2c_common_5411x.h.

#### 6.52.2.73 #define I2C_INTSTAT_SLVDESEL (1 << 15)

Slave Deselect Interrupt Status Bit

Definition at line 300 of file i2c_common_5411x.h.

#### 6.52.2.74 #define I2C_INTSTAT_SLVDESEL (1 << 15)

Slave Deselect Interrupt Status Bit

Definition at line 300 of file i2c_common_5411x.h.

#### 6.52.2.75 #define I2C_INTSTAT_SLVNOTSTR (1 << 11)

Slave not stretching Clock Interrupt Status Bit

Definition at line 299 of file i2c_common_5411x.h.

#### 6.52.2.76 #define I2C_INTSTAT_SLVNOTSTR (1 << 11)

Slave not stretching Clock Interrupt Status Bit

Definition at line 299 of file i2c_common_5411x.h.

#### 6.52.2.77 #define I2C_INTSTAT_SLVPENDING (1 << 8)

Slave Pending Interrupt Status Bit

Definition at line 298 of file i2c_common_5411x.h.

#### 6.52.2.78 #define I2C_INTSTAT_SLVPENDING (1 << 8)

Slave Pending Interrupt Status Bit

Definition at line 298 of file i2c_common_5411x.h.

#### 6.52.2.79 #define I2C_MONRXDAT_DATA (0xFF << 0)

Monitor Function Receive Data Field

Definition at line 354 of file i2c_common_5411x.h.

#### 6.52.2.80 #define I2C_MONRXDAT_DATA (0xFF << 0)

Monitor Function Receive Data Field

Definition at line 354 of file i2c_common_5411x.h.

#### 6.52.2.81 #define I2C_MONRXDAT_MONNACK (1 << 10)

Monitor Received Nack Bit

Definition at line 357 of file i2c_common_5411x.h.

#### 6.52.2.82 #define I2C_MONRXDAT_MONNACK (1 << 10)

Monitor Received Nack Bit

Definition at line 357 of file i2c_common_5411x.h.

#### 6.52.2.83 #define I2C_MONRXDAT_MONRESTART (1 << 9)

Monitor Received Repeated Start Bit

Definition at line 356 of file i2c_common_5411x.h.

#### 6.52.2.84 #define I2C_MONRXDAT_MONRESTART (1 << 9)

Monitor Received Repeated Start Bit

Definition at line 356 of file i2c_common_5411x.h.

#### 6.52.2.85 #define I2C_MONRXDAT_MONSTART (1 << 8)

Monitor Received Start Bit

Definition at line 355 of file i2c_common_5411x.h.

#### 6.52.2.86 #define I2C_MONRXDAT_MONSTART (1 << 8)

Monitor Received Start Bit

Definition at line 355 of file i2c_common_5411x.h.

#### 6.52.2.87 #define I2C_MSTCTL_MSTCONTINUE (1 << 0)

Master Continue Bit

Definition at line 310 of file i2c_common_5411x.h.

#### 6.52.2.88 #define I2C_MSTCTL_MSTCONTINUE (1 << 0)

Master Continue Bit

Definition at line 310 of file i2c_common_5411x.h.

#### 6.52.2.89 #define I2C_MSTCTL_MSTDMA (1 << 3)

Master DMA Enable Bit

Definition at line 313 of file i2c_common_5411x.h.

#### 6.52.2.90 #define I2C_MSTCTL_MSTDMA (1 << 3)

Master DMA Enable Bit

Definition at line 313 of file i2c_common_5411x.h.

#### 6.52.2.91 #define I2C_MSTCTL_MSTSTART (1 << 1)

Master Start Control Bit

Definition at line 311 of file i2c_common_5411x.h.

#### 6.52.2.92 #define I2C_MSTCTL_MSTSTART (1 << 1)

Master Start Control Bit

Definition at line 311 of file i2c_common_5411x.h.

#### 6.52.2.93 #define I2C_MSTCTL_MSTSTOP (1 << 2)

Master Stop Control Bit

Definition at line 312 of file i2c_common_5411x.h.

#### 6.52.2.94 #define I2C_MSTCTL_MSTSTOP (1 << 2)

Master Stop Control Bit

Definition at line 312 of file i2c_common_5411x.h.

#### 6.52.2.95 #define I2C_MSTDAT_DATAMASK ((uint32_t) 0x00FF << 0)

Master data mask

Definition at line 324 of file i2c_common_5411x.h.

#### 6.52.2.96 #define I2C_MSTDAT_DATAMASK ((uint32_t) 0x00FF << 0)

Master data mask

Definition at line 324 of file i2c_common_5411x.h.

#### 6.52.2.97 #define I2C_MSTTIME_MSTSCLHIGH (0x07 << 4)

Master SCL High Time field

Definition at line 319 of file i2c_common_5411x.h.

#### 6.52.2.98 #define I2C_MSTTIME_MSTSCLHIGH (0x07 << 4)

Master SCL High Time field

Definition at line 319 of file i2c_common_5411x.h.

#### 6.52.2.99 #define I2C_MSTTIME_MSTSCLLOW (0x07 << 0)

Master SCL Low Time field

Definition at line 318 of file i2c_common_5411x.h.

#### 6.52.2.100 #define I2C_MSTTIME_MSTSCLLOW (0x07 << 0)

Master SCL Low Time field

Definition at line 318 of file i2c_common_5411x.h.

#### 6.52.2.101 #define I2C_SLVADR_MASK ((uint32_t) 0x00FF)

Slave Address Mask

Definition at line 343 of file i2c_common_5411x.h.

#### 6.52.2.102 #define I2C_SLVADR_MASK ((uint32_t) 0x00FF)

Slave Address Mask

Definition at line 343 of file i2c_common_5411x.h.

#### 6.52.2.103 #define I2C_SLVADR_SADISABLE (1 << 0)

Slave Address n Disable Bit

Definition at line 341 of file i2c_common_5411x.h.

#### 6.52.2.104 #define I2C_SLVADR_SADISABLE (1 << 0)

Slave Address n Disable Bit

Definition at line 341 of file i2c_common_5411x.h.

#### 6.52.2.105 #define I2C_SLVADR_SLVADR (0x7F << 1)

Slave Address field

Definition at line 342 of file i2c_common_5411x.h.

#### 6.52.2.106 #define I2C_SLVADR_SLVADR (0x7F << 1)

Slave Address field

Definition at line 342 of file i2c_common_5411x.h.

#### 6.52.2.107 #define I2C_SLVCTL_SLVCONTINUE (1 << 0)

Slave Continue Bit

Definition at line 329 of file i2c_common_5411x.h.

### 6.52.2.108 #define I2C_SLVCTL_SLVCONTINUE (1 << 0)

Slave Continue Bit

Definition at line 329 of file i2c_common_5411x.h.

### 6.52.2.109 #define I2C_SLVCTL_SLVDMA (1 << 3)

Slave DMA Enable Bit

Definition at line 331 of file i2c_common_5411x.h.

### 6.52.2.110 #define I2C_SLVCTL_SLVDMA (1 << 3)

Slave DMA Enable Bit

Definition at line 331 of file i2c_common_5411x.h.

### 6.52.2.111 #define I2C_SLVCTL_SLVNACK (1 << 1)

Slave NACK Bit

Definition at line 330 of file i2c_common_5411x.h.

### 6.52.2.112 #define I2C_SLVCTL_SLVNACK (1 << 1)

Slave NACK Bit

Definition at line 330 of file i2c_common_5411x.h.

### 6.52.2.113 #define I2C_SLVDAT_DATAMASK ((uint32_t) 0x00FF << 0)

Slave data mask

Definition at line 336 of file i2c_common_5411x.h.

### 6.52.2.114 #define I2C_SLVDAT_DATAMASK ((uint32_t) 0x00FF << 0)

Slave data mask

Definition at line 336 of file i2c_common_5411x.h.

### 6.52.2.115 #define I2C_SLVQUAL_QUALMODE0 (1 << 0)

Slave Qualifier Mode Enable Bit

Definition at line 348 of file i2c_common_5411x.h.

### 6.52.2.116 #define I2C_SLVQUAL_QUALMODE0 (1 << 0)

Slave Qualifier Mode Enable Bit

Definition at line 348 of file i2c_common_5411x.h.

#### 6.52.2.117 #define I2C_SLVQUAL_SLVQUAL0 (0x7F << 1)

Slave Qualifier Address for Address 0

Definition at line 349 of file i2c_common_5411x.h.

#### 6.52.2.118 #define I2C_SLVQUAL_SLVQUAL0 (0x7F << 1)

Slave Qualifier Address for Address 0

Definition at line 349 of file i2c_common_5411x.h.

#### 6.52.2.119 #define I2C_STAT_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Flag

Definition at line 244 of file i2c_common_5411x.h.

#### 6.52.2.120 #define I2C_STAT_EVENTTIMEOUT (1 << 24)

Event Timeout Interrupt Flag

Definition at line 244 of file i2c_common_5411x.h.

#### 6.52.2.121 #define I2C_STAT_MONACTIVE (1 << 18)

Monitor Active Flag

Definition at line 242 of file i2c_common_5411x.h.

#### 6.52.2.122 #define I2C_STAT_MONACTIVE (1 << 18)

Monitor Active Flag

Definition at line 242 of file i2c_common_5411x.h.

#### 6.52.2.123 #define I2C_STAT_MONIDLE (1 << 19)

Monitor Idle Flag

Definition at line 243 of file i2c_common_5411x.h.

#### 6.52.2.124 #define I2C_STAT_MONIDLE (1 << 19)

Monitor Idle Flag

Definition at line 243 of file i2c_common_5411x.h.

#### 6.52.2.125 #define I2C_STAT_MONOV (1 << 17)

Monitor Overflow Flag

Definition at line 241 of file i2c_common_5411x.h.

#### 6.52.2.126 #define I2C_STAT_MONOV (1 << 17)

Monitor Overflow Flag

Definition at line 241 of file i2c_common_5411x.h.

#### 6.52.2.127 #define I2C_STAT_MONRDY (1 << 16)

Monitor Ready Bit

Definition at line 240 of file i2c_common_5411x.h.

#### 6.52.2.128 #define I2C_STAT_MONRDY (1 << 16)

Monitor Ready Bit

Definition at line 240 of file i2c_common_5411x.h.

#### 6.52.2.129 #define I2C_STAT_MSTCODE_IDLE (0)

Master Idle State Code

Definition at line 247 of file i2c_common_5411x.h.

#### 6.52.2.130 #define I2C_STAT_MSTCODE_IDLE (0)

Master Idle State Code

Definition at line 247 of file i2c_common_5411x.h.

#### 6.52.2.131 #define I2C_STAT_MSTCODE_NACKADR (3)

Master NACK by slave on address State Code

Definition at line 250 of file i2c_common_5411x.h.

#### 6.52.2.132 #define I2C_STAT_MSTCODE_NACKADR (3)

Master NACK by slave on address State Code

Definition at line 250 of file i2c_common_5411x.h.

#### 6.52.2.133 #define I2C_STAT_MSTCODE_NACKDAT (4)

Master NACK by slave on data State Code

Definition at line 251 of file i2c_common_5411x.h.

#### 6.52.2.134 #define I2C_STAT_MSTCODE_NACKDAT (4)

Master NACK by slave on data State Code

Definition at line 251 of file i2c_common_5411x.h.

#### 6.52.2.135 #define I2C_STAT_MSTCODE_RXREADY (1)

Master Receive Ready State Code

Definition at line 248 of file i2c_common_5411x.h.

#### 6.52.2.136 #define I2C_STAT_MSTCODE_RXREADY (1)

Master Receive Ready State Code

Definition at line 248 of file i2c_common_5411x.h.

#### 6.52.2.137 #define I2C_STAT_MSTCODE_TXREADY (2)

Master Transmit Ready State Code

Definition at line 249 of file i2c_common_5411x.h.

#### 6.52.2.138 #define I2C_STAT_MSTCODE_TXREADY (2)

Master Transmit Ready State Code

Definition at line 249 of file i2c_common_5411x.h.

#### 6.52.2.139 #define I2C_STAT_MSTPENDING (1 << 0)

Master Pending Status Bit

Definition at line 230 of file i2c_common_5411x.h.

#### 6.52.2.140 #define I2C_STAT_MSTPENDING (1 << 0)

Master Pending Status Bit

Definition at line 230 of file i2c_common_5411x.h.

#### 6.52.2.141 #define I2C_STAT_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Bit

Definition at line 232 of file i2c_common_5411x.h.

#### 6.52.2.142 #define I2C_STAT_MSTRARBLOSS (1 << 4)

Master Arbitration Loss Bit

Definition at line 232 of file i2c_common_5411x.h.

#### 6.52.2.143 #define I2C_STAT_MSTSTATE (0x7 << 1)

Master State Code

Definition at line 231 of file i2c_common_5411x.h.

#### 6.52.2.144 #define I2C_STAT_MSTSTATE (0x7 << 1)

Master State Code

Definition at line 231 of file i2c_common_5411x.h.

#### 6.52.2.145 #define I2C_STAT_MSTSTSTPERR (1 << 6)

Master Start Stop Error Bit

Definition at line 233 of file i2c_common_5411x.h.

#### 6.52.2.146 #define I2C_STAT_MSTSTSTPERR (1 << 6)

Master Start Stop Error Bit

Definition at line 233 of file i2c_common_5411x.h.

#### 6.52.2.147 #define I2C_STAT_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Flag

Definition at line 245 of file i2c_common_5411x.h.

#### 6.52.2.148 #define I2C_STAT_SCLTIMEOUT (1 << 25)

SCL Timeout Interrupt Flag

Definition at line 245 of file i2c_common_5411x.h.

#### 6.52.2.149 #define I2C_STAT_SLVCODE_ADDR (0)

Master Idle State Code

Definition at line 253 of file i2c_common_5411x.h.

#### 6.52.2.150 #define I2C_STAT_SLVCODE_ADDR (0)

Master Idle State Code

Definition at line 253 of file i2c_common_5411x.h.

#### 6.52.2.151 #define I2C_STAT_SLVCODE_RX (1)

Received data is available Code

Definition at line 254 of file i2c_common_5411x.h.

#### 6.52.2.152 #define I2C_STAT_SLVCODE_RX (1)

Received data is available Code

Definition at line 254 of file i2c_common_5411x.h.

#### 6.52.2.153 #define I2C_STAT_SLVCODE_TX (2)

Data can be transmitted Code

Definition at line 255 of file i2c_common_5411x.h.

#### 6.52.2.154 #define I2C_STAT_SLVCODE_TX (2)

Data can be transmitted Code

Definition at line 255 of file i2c_common_5411x.h.

#### 6.52.2.155 #define I2C_STAT_SLVDESEL (1 << 15)

Slave Deselect Bit

Definition at line 239 of file i2c_common_5411x.h.

#### 6.52.2.156 #define I2C_STAT_SLVDESEL (1 << 15)

Slave Deselect Bit

Definition at line 239 of file i2c_common_5411x.h.

#### 6.52.2.157 #define I2C_STAT_SLVIDX (0x3 << 12)

Slave Address Index

Definition at line 237 of file i2c_common_5411x.h.

#### 6.52.2.158 #define I2C_STAT_SLVIDX (0x3 << 12)

Slave Address Index

Definition at line 237 of file i2c_common_5411x.h.

#### 6.52.2.159 #define I2C_STAT_SLVNOTSTR (1 << 11)

Slave not stretching Clock Bit

Definition at line 236 of file i2c_common_5411x.h.

#### 6.52.2.160 #define I2C_STAT_SLVNOTSTR (1 << 11)

Slave not stretching Clock Bit

Definition at line 236 of file i2c_common_5411x.h.

#### 6.52.2.161 #define I2C_STAT_SLVPENDING (1 << 8)

Slave Pending Status Bit

Definition at line 234 of file i2c_common_5411x.h.

#### 6.52.2.162 #define I2C_STAT_SLVPENDING (1 $<<$ 8)

Slave Pending Status Bit

Definition at line 234 of file i2c_common_5411x.h.

#### 6.52.2.163 #define I2C_STAT_SLVSEL (1 $<<$ 14)

Slave Selected Bit

Definition at line 238 of file i2c_common_5411x.h.

#### 6.52.2.164 #define I2C_STAT_SLVSEL (1 $<<$ 14)

Slave Selected Bit

Definition at line 238 of file i2c_common_5411x.h.

#### 6.52.2.165 #define I2C_STAT_SLVSTATE (0x3 $<<$ 9)

Slave State Code

Definition at line 235 of file i2c_common_5411x.h.

#### 6.52.2.166 #define I2C_STAT_SLVSTATE (0x3 $<<$ 9)

Slave State Code

Definition at line 235 of file i2c_common_5411x.h.

#### 6.52.2.167 #define I2C_TIMEOUT_VAL( *n* ) (((uint32_t) ((n) - 1) & 0xFFF0) | 0x000F)

Macro for Timeout value register

Definition at line 290 of file i2c_common_5411x.h.

#### 6.52.2.168 #define I2C_TIMEOUT_VAL( *n* ) (((uint32_t) ((n) - 1) & 0xFFF0) | 0x000F)

Macro for Timeout value register

Definition at line 290 of file i2c_common_5411x.h.

### 6.52.3 Function Documentation

#### 6.52.3.1 static INLINE void Chip_I2C_ClearInt ( LPC_I2C_T ∗ *pI2C,* uint32_t *intClr* ) [static]

Disable I2C Interrupts.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *intClr* | : ORed Value of I2C_INTENSET_∗ values to disable |

**Returns**

Nothing

**Note**

It is recommended to use the Chip_I2C_DisableInt() function instead of this function.

Definition at line 447 of file i2c_common_5411x.h.

#### 6.52.3.2 __STATIC_INLINE void Chip_I2C_DeInit ( LPC_I2C_T ∗ *pI2C* )

Shutdown I2C Interface.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Nothing

**Note**

This function disables the I2C clock for both the master and slave interfaces if the I2C channel.

Definition at line 379 of file i2c_common_5411x.h.

#### 6.52.3.3 static INLINE void Chip_I2C_DisableInt ( LPC_I2C_T ∗ *pI2C,* uint32_t *intClr* ) [static]

Disable I2C Interrupts.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *intClr* | : ORed Value of I2C_INTENSET_∗ values to disable |

**Returns**

Nothing

Definition at line 434 of file i2c_common_5411x.h.

#### 6.52.3.4 static INLINE void Chip_I2C_EnableInt ( LPC_I2C_T ∗ *pI2C,* uint32_t *intEn* ) [static]

Enable I2C Interrupts.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |
| *intEn* | : ORed Value of I2C_INTENSET_∗ values to enable |

**Returns**

Nothing

Definition at line 423 of file i2c_common_5411x.h.

#### 6.52.3.5 static INLINE uint32_t Chip_I2C_GetClockDiv ( LPC_I2C_T ∗ *pI2C* ) [static]

Get I2C Clock Divider registers.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

Clock Divider value

**Note**

Return the divider value for the I2C block It is the CLKDIV register value + 1

Definition at line 412 of file i2c_common_5411x.h.

#### 6.52.3.6 static INLINE uint32_t Chip_I2C_GetPendingInt ( LPC_I2C_T ∗ *pI2C* ) [static]

Returns pending I2C Interrupts.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

All pending interrupts, mask with I2C_INTENSET_∗ to determine specific interrupts

Definition at line 457 of file i2c_common_5411x.h.

#### 6.52.3.7 __STATIC_INLINE int Chip_I2C_Init ( LPC_I2C_T ∗ *pI2C* )

Initialize I2C Interface.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

**Returns**

0 - on success; ERR_FLEXCOMM_FUNCNOTSUPPORTED or ERR_FLEXCOMM_NOTFREE on failure

**Note**

This function enables the I2C clock for both the master and slave interfaces if the I2C channel.

Definition at line 367 of file i2c_common_5411x.h.

#### 6.52.3.8 __STATIC_INLINE void Chip_I2C_SetClockDiv ( LPC_I2C_T ∗ *pI2C,* uint32_t *clkdiv* )

Sets I2C Clock Divider registers.

**Parameters**

| | |
|---|---|
| *pI2C* | : Pointer to selected I2C peripheral |

|  |  |
|---|---|
| *clkdiv* | : Clock Divider value for I2C, value is between (1 - 65536) |

**Returns**

Nothing

**Note**

The clock to I2C block is determined by the following formula (I2C_PCLK is the frequency of the system clock):
I2C Clock Frequency = (I2C_PCLK)/clkdiv;
This divider must be setup for both the master and slave modes of the controller.

Definition at line 395 of file i2c_common_5411x.h.

## 6.52.4 Variable Documentation

### 6.52.4.1 __IO uint32_t CLKDIV

I2C Clock Divider Register

Definition at line 55 of file i2c_common_5411x.h.

### 6.52.4.2 __O uint32_t INTENCLR

I2C Interrupt Enable Clear Register common for Master, Slave and Monitor

Definition at line 53 of file i2c_common_5411x.h.

### 6.52.4.3 __IO uint32_t INTENSET

I2C Interrupt Enable Set Register common for Master, Slave and Monitor

Definition at line 52 of file i2c_common_5411x.h.

### 6.52.4.4 __IO uint32_t INTSTAT

I2C Interrupt Status Register

Definition at line 56 of file i2c_common_5411x.h.

### 6.52.4.5 LPC_I2C_T

Definition at line 73 of file i2c_common_5411x.h.

### 6.52.4.6 __I uint32_t MONRXDAT

I2C Monitor Data Register

Definition at line 67 of file i2c_common_5411x.h.

### 6.52.4.7 __IO uint32_t MSTCTL

I2C Master Control Register

Definition at line 58 of file i2c_common_5411x.h.

#### 6.52.4.8 __IO uint32_t MSTDAT

I2C Master Data Register

Definition at line 60 of file i2c_common_5411x.h.

#### 6.52.4.9 __IO uint32_t MSTTIME

I2C Master Time Register for SCL

Definition at line 59 of file i2c_common_5411x.h.

#### 6.52.4.10 __I uint32_t PID

Offset: 0xFFC Module identification register

Definition at line 72 of file i2c_common_5411x.h.

#### 6.52.4.11 __IO uint32_t PSELID

Offset: 0xFF8 Peripheral select/identification register

Definition at line 71 of file i2c_common_5411x.h.

#### 6.52.4.12 __I uint32_t RESERVED0

Definition at line 57 of file i2c_common_5411x.h.

#### 6.52.4.13 __IO uint32_t RESERVED1[5]

Definition at line 61 of file i2c_common_5411x.h.

#### 6.52.4.14 __IO uint32_t RESERVED2[9]

Definition at line 66 of file i2c_common_5411x.h.

#### 6.52.4.15 __IO uint32_t SLVADR[4]

I2C Slave Address Registers

Definition at line 64 of file i2c_common_5411x.h.

#### 6.52.4.16 __IO uint32_t SLVCTL

I2C Slave Control Register

Definition at line 62 of file i2c_common_5411x.h.

#### 6.52.4.17 __IO uint32_t SLVDAT

I2C Slave Data Register

Definition at line 63 of file i2c_common_5411x.h.

#### 6.52.4.18 __IO uint32_t SLVQUAL0

I2C Slave Address Qualifier 0 Register

Definition at line 65 of file i2c_common_5411x.h.

#### 6.52.4.19 __IO uint32_t STAT

I2C register block structure.

I2C Configuration Register common for Master, Slave and Monitor I2C Status Register common for Master, Slave and Monitor

Definition at line 51 of file i2c_common_5411x.h.

#### 6.52.4.20 __IO uint32_t TIMEOUT

I2C Timeout value Register

Definition at line 54 of file i2c_common_5411x.h.

## 6.53 CHIP: RTC tick to (a more) Universal Time conversion functions

### 6.53.1 Detailed Description

This driver converts between a RTC 1-second tick value and a Universal time format in a structure of type 'struct tm'.

### Macros

- #define TM_YEAR_BASE (1970)
- #define TM_DAYOFWEEK (4)

### Functions

- void ConvertRtcTime (uint32_t rtcTick, struct tm ∗pTime)

  *Converts a RTC tick time to Universal time.*
- void ConvertTimeRtc (struct tm ∗pTime, uint32_t ∗rtcTick)

  *Converts a Universal time to RTC tick time.*

### 6.53.2 Macro Definition Documentation

#### 6.53.2.1 #define TM_DAYOFWEEK (4)

Definition at line 54 of file rtc_ut.h.

#### 6.53.2.2 #define TM_YEAR_BASE (1970)

Definition at line 53 of file rtc_ut.h.

### 6.53.3 Function Documentation

#### 6.53.3.1 void ConvertRtcTime ( uint32_t *rtcTick,* struct tm ∗ *pTime* )

Converts a RTC tick time to Universal time.

**Parameters**

| | |
|---|---|
| *rtcTick* | : Current RTC time value |
| *pTime* | : Pointer to time structure to fill |

**Returns**

Nothing

**Note**

When setting time, the 'tm_wday', 'tm_yday', and 'tm_isdst' fields are not used.

Definition at line 130 of file rtc_ut.c.

#### 6.53.3.2 void ConvertTimeRtc ( struct tm ∗ *pTime,* uint32_t ∗ *rtcTick* )

Converts a Universal time to RTC tick time.

**Parameters**

| | |
|---|---|
| *pTime* | : Pointer to time structure to use |
| *rtcTick* | : Pointer to RTC time value to fill |

**Returns**

Nothing

**Note**

When converting time, the 'tm_isdst' field is not populated by the conversion function.

Definition at line 154 of file rtc_ut.c.

# 6.54 CHIP: Simple ring buffer implementation

## 6.54.1 Detailed Description

### Data Structures

- struct RINGBUFF_T

  *Ring buffer structure.*

### Macros

- #define RB_VHEAD(rb) (∗(volatile uint32_t ∗) &(rb)->head)
- #define RB_VTAIL(rb) (∗(volatile uint32_t ∗) &(rb)->tail)

### Functions

- int RingBuffer_Init (RINGBUFF_T ∗RingBuff, void ∗buffer, int itemSize, int count, void ∗(∗cpyFunc)(void ∗dst, const void ∗src, uint32_t len))

  *Initialize ring buffer.*
- __STATIC_INLINE void RingBuffer_Flush (RINGBUFF_T ∗RingBuff)

  *Resets the ring buffer to empty.*
- __STATIC_INLINE int RingBuffer_GetSize (RINGBUFF_T ∗RingBuff)

  *Return size the ring buffer.*
- __STATIC_INLINE int RingBuffer_GetCount (RINGBUFF_T ∗RingBuff)

  *Return number of items in the ring buffer.*
- __STATIC_INLINE int RingBuffer_GetFree (RINGBUFF_T ∗RingBuff)

  *Return number of free items in the ring buffer.*
- __STATIC_INLINE int RingBuffer_IsFull (RINGBUFF_T ∗RingBuff)

  *Return number of items in the ring buffer.*
- __STATIC_INLINE int RingBuffer_IsEmpty (RINGBUFF_T ∗RingBuff)

  *Return empty status of ring buffer.*
- int RingBuffer_Insert (RINGBUFF_T ∗RingBuff, const void ∗data)

  *Insert a single item into ring buffer.*
- int RingBuffer_InsertMult (RINGBUFF_T ∗RingBuff, const void ∗data, int num)

  *Insert an array of items into ring buffer.*
- int RingBuffer_Pop (RINGBUFF_T ∗RingBuff, void ∗data)

  *Pop an item from the ring buffer.*
- int RingBuffer_PopMult (RINGBUFF_T ∗RingBuff, void ∗data, int num)

  *Pop an array of items from the ring buffer.*

## 6.54.2 Macro Definition Documentation

### 6.54.2.1 #define RB_VHEAD( *rb* ) (∗(volatile uint32_t ∗) &(rb)->head)

volatile typecasted head index

Definition at line 59 of file ring_buffer.h.

### 6.54.2.2 #define RB_VTAIL( *rb* ) (∗(volatile uint32_t ∗) &(rb)->tail)

volatile typecasted tail index

Definition at line 65 of file ring_buffer.h.

## 6.54.3 Function Documentation

### 6.54.3.1 __STATIC_INLINE void RingBuffer_Flush ( RINGBUFF_T ∗ *RingBuff* )

Resets the ring buffer to empty.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |

**Returns**

Nothing

Definition at line 90 of file ring_buffer.h.

### 6.54.3.2 __STATIC_INLINE int RingBuffer_GetCount ( RINGBUFF_T ∗ *RingBuff* )

Return number of items in the ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |

**Returns**

Number of items in the ring buffer

Definition at line 110 of file ring_buffer.h.

### 6.54.3.3 __STATIC_INLINE int RingBuffer_GetFree ( RINGBUFF_T ∗ *RingBuff* )

Return number of free items in the ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |

**Returns**

Number of free items in the ring buffer

Definition at line 120 of file ring_buffer.h.

### 6.54.3.4 __STATIC_INLINE int RingBuffer_GetSize ( RINGBUFF_T ∗ *RingBuff* )

Return size the ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |

**Returns**

Size of the ring buffer in bytes

Definition at line 100 of file ring_buffer.h.

#### 6.54.3.5 int RingBuffer_Init ( RINGBUFF_T ∗ *RingBuff,* void ∗ *buffer,* int *itemSize,* int *count,* void ∗(∗)(void ∗dst, const void ∗src, uint32_t len) *cpyFunc* )

Initialize ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer to initialize |
| *buffer* | : Pointer to buffer to associate with RingBuff |
| *itemSize* | : Size of each buffer item size |
| *count* | : Size of ring buffer |
| *cpyFunc* | : Call-back function that copies data (if NULL library *memcpy* will be used) |

**Note**

Memory pointed by *buffer* must have correct alignment of *itemSize*, and *count* must be a power of 2 and must at least be 2 or greater. *len* of the *cpyFunc* is in bytes.

**Returns**

Nothing

Definition at line 55 of file ring_buffer.c.

#### 6.54.3.6 int RingBuffer_Insert ( RINGBUFF_T ∗ *RingBuff,* const void ∗ *data* )

Insert a single item into ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |
| *data* | : pointer to item |

**Returns**

1 when successfully inserted, 0 on error (Buffer not initialized using RingBuffer_Init() or attempted to insert when buffer is full)

Definition at line 74 of file ring_buffer.c.

#### 6.54.3.7 int RingBuffer_InsertMult ( RINGBUFF_T ∗ *RingBuff,* const void ∗ *data,* int *num* )

Insert an array of items into ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |
| *data* | : Pointer to first element of the item array |
| *num* | : Number of items in the array |

**Returns**

number of items successfully inserted, 0 on error (Buffer not initialized using RingBuffer_Init() or attempted to insert when buffer is full)

Definition at line 91 of file ring_buffer.c.

#### 6.54.3.8 __STATIC_INLINE int RingBuffer_IsEmpty ( RINGBUFF_T ∗ *RingBuff* )

Return empty status of ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |

**Returns**

1 if the ring buffer is empty, otherwise 0

Definition at line 140 of file ring_buffer.h.

#### 6.54.3.9 __STATIC_INLINE int RingBuffer_IsFull ( RINGBUFF_T ∗ *RingBuff* )

Return number of items in the ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |

**Returns**

1 if the ring buffer is full, otherwise 0

Definition at line 130 of file ring_buffer.h.

#### 6.54.3.10 int RingBuffer_Pop ( RINGBUFF_T ∗ *RingBuff,* void ∗ *data* )

Pop an item from the ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |
| *data* | : Pointer to memory where popped item be stored |

**Returns**

1 when item popped successfuly onto *data*, 0 When error (Buffer not initialized using RingBuffer_Init() or attempted to pop item when the buffer is empty)

Definition at line 129 of file ring_buffer.c.

#### 6.54.3.11 int RingBuffer_PopMult ( RINGBUFF_T ∗ *RingBuff,* void ∗ *data,* int *num* )

Pop an array of items from the ring buffer.

**Parameters**

| | |
|---|---|
| *RingBuff* | : Pointer to ring buffer |
| *data* | : Pointer to memory where popped items be stored |
| *num* | : Max number of items array *data* can hold |

**Returns**

Number of items popped onto *data*, 0 on error (Buffer not initialized using RingBuffer_Init() or attempted to pop when the buffer is empty)

Definition at line 146 of file ring_buffer.c.

# 6.55 CHIP: Stopwatch primitives.

## 6.55.1 Detailed Description

### Functions

- void StopWatch_Init (void)

  *Initialize stopwatch.*
- uint32_t StopWatch_Start (void)

  *Start a stopwatch.*
- __STATIC_INLINE uint32_t StopWatch_Elapsed (uint32_t startTime)

  *Returns number of ticks elapsed since stopwatch was started.*
- uint32_t StopWatch_TicksPerSecond (void)

  *Returns number of ticks per second of the stopwatch timer.*
- uint32_t StopWatch_TicksToMs (uint32_t ticks)

  *Converts from stopwatch ticks to mS.*
- uint32_t StopWatch_TicksToUs (uint32_t ticks)

  *Converts from stopwatch ticks to uS.*
- uint32_t StopWatch_MsToTicks (uint32_t mS)

  *Converts from mS to stopwatch ticks.*
- uint32_t StopWatch_UsToTicks (uint32_t uS)

  *Converts from uS to stopwatch ticks.*
- __STATIC_INLINE void StopWatch_DelayTicks (uint32_t ticks)

  *Delays the given number of ticks using stopwatch primitives.*
- __STATIC_INLINE void StopWatch_DelayMs (uint32_t mS)

  *Delays the given number of mS using stopwatch primitives.*
- __STATIC_INLINE void StopWatch_DelayUs (uint32_t uS)

  *Delays the given number of uS using stopwatch primitives.*

## 6.55.2 Function Documentation

### 6.55.2.1 __STATIC_INLINE void StopWatch_DelayMs ( uint32_t *mS* )

Delays the given number of mS using stopwatch primitives.

**Parameters**

| | |
|---|---|
| *mS* | : Number of mS to delay |

**Returns**

Nothing

Definition at line 114 of file stopwatch.h.

### 6.55.2.2 __STATIC_INLINE void StopWatch_DelayTicks ( uint32_t *ticks* )

Delays the given number of ticks using stopwatch primitives.

**Parameters**

| | |
|---|---|
| *ticks* | : Number of ticks to delay |

**Returns**

Nothing

Definition at line 103 of file stopwatch.h.

### 6.55.2.3 __STATIC_INLINE void StopWatch_DelayUs ( uint32_t *uS* )

Delays the given number of uS using stopwatch primitives.

**Parameters**

| | |
|---|---|
| *uS* | : Number of uS to delay |

**Returns**

Nothing

Definition at line 126 of file stopwatch.h.

### 6.55.2.4 __STATIC_INLINE uint32_t StopWatch_Elapsed ( uint32_t *startTime* )

Returns number of ticks elapsed since stopwatch was started.

**Parameters**

| | |
|---|---|
| *startTime* | : Time returned by StopWatch_Start(). |

**Returns**

Number of ticks elapsed since stopwatch was started

Definition at line 59 of file stopwatch.h.

### 6.55.2.5 void StopWatch_Init ( void )

Initialize stopwatch.

**Returns**

Nothing

Definition at line 60 of file stopwatch_5411x.c.

### 6.55.2.6 uint32_t StopWatch_MsToTicks ( uint32_t *mS* )

Converts from mS to stopwatch ticks.

**Parameters**

| | |
|---|---|
| *mS* | : Duration in mS to convert to ticks. |

**Returns**

Number of ticks in given number of mS

Definition at line 100 of file stopwatch_5411x.c.

#### 6.55.2.7 uint32_t StopWatch_Start ( void )

Start a stopwatch.

**Returns**

Current cycle count

Definition at line 75 of file stopwatch_5411x.c.

#### 6.55.2.8 uint32_t StopWatch_TicksPerSecond ( void )

Returns number of ticks per second of the stopwatch timer.

**Returns**

Number of ticks per second of the stopwatch timer

Definition at line 82 of file stopwatch_5411x.c.

#### 6.55.2.9 uint32_t StopWatch_TicksToMs ( uint32_t *ticks* )

Converts from stopwatch ticks to mS.

**Parameters**

| | |
|---|---|
| *ticks* | : Duration in ticks to convert to mS. |

**Returns**

Number of mS in given number of ticks

Definition at line 88 of file stopwatch_5411x.c.

#### 6.55.2.10 uint32_t StopWatch_TicksToUs ( uint32_t *ticks* )

Converts from stopwatch ticks to uS.

**Parameters**

| | |
|---|---|
| *ticks* | : Duration in ticks to convert to uS. |

**Returns**

Number of uS in given number of ticks

Definition at line 94 of file stopwatch_5411x.c.

#### 6.55.2.11 uint32_t StopWatch_UsToTicks ( uint32_t *uS* )

Converts from uS to stopwatch ticks.

Parameters

| | |
|---|---|
| *uS* | : Duration in uS to convert to ticks. |

Returns

Number of ticks in given number of uS

Definition at line 106 of file stopwatch_5411x.c.

# 6.56 CHIP_5411X: LPC5411X M0 core peripheral interrupt numbers

## 6.56.1 Detailed Description

### Enumerations

- enum LPC5411X_M0_IRQn_Type {
  Reset_IRQn = -15, NonMaskableInt_IRQn = -14, HardFault_IRQn = -13, SVCall_IRQn = -5,
  PendSV_IRQn = -2, SysTick_IRQn = -1, WDTBOD_IRQn, DMA_IRQn,
  GINT0_IRQn, GINT1_IRQn, PIN_INT0_IRQn, PIN_INT1_IRQn,
  PIN_INT2_IRQn, PIN_INT3_IRQn, UTICK_IRQn, MRT_IRQn,
  CT32B0_IRQn, CT32B1_IRQn, SCT0_IRQn, CT32B3_IRQn,
  FLEXCOMM0_IRQn, FLEXCOMM1_IRQn, FLEXCOMM2_IRQn, FLEXCOMM3_IRQn,
  FLEXCOMM4_IRQn, FLEXCOMM5_IRQn, FLEXCOMM6_IRQn, FLEXCOMM7_IRQn,
  ADC_SEQA_IRQn, ADC_SEQB_IRQn, ADC_THCMP_IRQn, DMIC_IRQn,
  HWVAD, USBACT_IRQn, USB_IRQn, RTC_IRQn,
  Reserved_IRQn, MAILBOX_IRQn }

## 6.56.2 Enumeration Type Documentation

### 6.56.2.1 enum LPC5411X_M0_IRQn_Type

#### Enumerator

***Reset_IRQn*** 1 Reset Vector, invoked on Power up and warm reset

***NonMaskableInt_IRQn*** 2 Non Maskable Interrupt

***HardFault_IRQn*** 3 Cortex-M0 Hard Fault Interrupt

***SVCall_IRQn*** 11 Cortex-M0 SV Call Interrupt

***PendSV_IRQn*** 14 Cortex-M0 Pend SV Interrupt

***SysTick_IRQn*** 15 Cortex-M0 System Tick Interrupt

***WDTBOD_IRQn*** WWDT

***DMA_IRQn*** DMA

***GINT0_IRQn*** GINT0

***GINT1_IRQn*** GINT1

***PIN_INT0_IRQn*** PININT0

***PIN_INT1_IRQn*** PININT1

***PIN_INT2_IRQn*** PININT2

***PIN_INT3_IRQn*** PININT3

***UTICK_IRQn*** Micro-tick Timer interrupt

***MRT_IRQn*** Multi-rate timer interrupt

***CT32B0_IRQn*** CTMR0

***CT32B1_IRQn*** CTMR1

***SCT0_IRQn*** SCT

***CT32B3_IRQn*** CTMR3

***FLEXCOMM0_IRQn*** FLEXCOMM0

***FLEXCOMM1_IRQn*** FLEXCOMM1

***FLEXCOMM2_IRQn*** FLEXCOMM2

***FLEXCOMM3_IRQn*** FLEXCOMM3

***FLEXCOMM4_IRQn*** FLEXCOMM4

***FLEXCOMM5_IRQn*** FLEXCOMM5

***FLEXCOMM6_IRQn*** FLEXCOMM6

***FLEXCOMM7_IRQn*** FLEXCOMM7

***ADC_SEQA_IRQn*** ADC0 sequence A completion

***ADC_SEQB_IRQn*** ADC0 sequence B completion

***ADC_THCMP_IRQn*** ADC0 threshold compare and error

***DMIC_IRQn*** Digital Mic

***HWVAD*** Hardware Voice acitivity detect

***USBACT_IRQn*** USB Activity

***USB_IRQn*** USB

***RTC_IRQn*** RTC alarm and wake-up interrupts

***Reserved_IRQn*** Reserved Interrupt

***MAILBOX_IRQn*** Mailbox

Definition at line 77 of file cmsis_5411x_m0.h.

# 6.57 CHIP_5411X: LPC5411X M4 core peripheral interrupt numbers

## 6.57.1 Detailed Description

### Enumerations

- enum LPC5411X_IRQn_Type {
Reset_IRQn = -15, NonMaskableInt_IRQn = -14, HardFault_IRQn = -13, MemoryManagement_IRQn = -12,
BusFault_IRQn = -11, UsageFault_IRQn = -10, SVCall_IRQn = -5, DebugMonitor_IRQn = -4,
PendSV_IRQn = -2, SysTick_IRQn = -1, WDTBOD_IRQn, DMA_IRQn,
GINT0_IRQn, GINT1_IRQn, PIN_INT0_IRQn, PIN_INT1_IRQn,
PIN_INT2_IRQn, PIN_INT3_IRQn, UTICK_IRQn, MRT_IRQn,
CT32B0_IRQn, CT32B1_IRQn, SCT0_IRQn, CT32B3_IRQn,
FLEXCOMM0_IRQn, FLEXCOMM1_IRQn, FLEXCOMM2_IRQn, FLEXCOMM3_IRQn,
FLEXCOMM4_IRQn, FLEXCOMM5_IRQn, FLEXCOMM6_IRQn, FLEXCOMM7_IRQn,
ADC_SEQA_IRQn, ADC_SEQB_IRQn, ADC_THCMP_IRQn, DMIC_IRQn,
HWVAD_IRQn, USBACT_IRQn, USB_IRQn, RTC_IRQn,
Reserved_IRQn, MAILBOX_IRQn, PIN_INT4_IRQn, PIN_INT5_IRQn,
PIN_INT6_IRQn, PIN_INT7_IRQn, CT32B2_IRQn, CT32B4_IRQn,
Reserved1_IRQn, SPIFI_IRQn }

## 6.57.2 Enumeration Type Documentation

### 6.57.2.1 enum LPC5411X_IRQn_Type

**Enumerator**

***Reset_IRQn*** 1 Reset Vector, invoked on Power up and warm reset

***NonMaskableInt_IRQn*** 2 Non maskable Interrupt, cannot be stopped or preempted

***HardFault_IRQn*** 3 Hard Fault, all classes of Fault

***MemoryManagement_IRQn*** 4 Memory Management, MPU mismatch, including Access Violation and No Match

***BusFault_IRQn*** 5 Bus Fault, Pre-Fetch-, Memory Access Fault, other address/memory related Fault

***UsageFault_IRQn*** 6 Usage Fault, i.e. Undef Instruction, Illegal State Transition

***SVCall_IRQn*** 11 System Service Call via SVC instruction

***DebugMonitor_IRQn*** 12 Debug Monitor

***PendSV_IRQn*** 14 Pendable request for system service

***SysTick_IRQn*** 15 System Tick Timer

***WDTBOD_IRQn*** WWDT

***DMA_IRQn*** DMA

***GINT0_IRQn*** GINT0

***GINT1_IRQn*** GINT1

***PIN_INT0_IRQn*** PININT0

***PIN_INT1_IRQn*** PININT1

***PIN_INT2_IRQn*** PININT2

***PIN_INT3_IRQn*** PININT3

***UTICK_IRQn*** Micro-tick Timer interrupt

***MRT_IRQn*** Multi-rate timer interrupt

***CT32B0_IRQn*** CTMR0

***CT32B1_IRQn*** CTMR1

***SCT0_IRQn*** SCT

***CT32B3_IRQn*** CTMR3

***FLEXCOMM0_IRQn*** FLEXCOMM0

***FLEXCOMM1_IRQn*** FLEXCOMM1

***FLEXCOMM2_IRQn*** FLEXCOMM2

***FLEXCOMM3_IRQn*** FLEXCOMM3

***FLEXCOMM4_IRQn*** FLEXCOMM4

***FLEXCOMM5_IRQn*** FLEXCOMM5

***FLEXCOMM6_IRQn*** FLEXCOMM6

***FLEXCOMM7_IRQn*** FLEXCOMM7

***ADC_SEQA_IRQn*** ADC0 sequence A completion

***ADC_SEQB_IRQn*** ADC0 sequence B completion

***ADC_THCMP_IRQn*** ADC0 threshold compare and error

***DMIC_IRQn*** Digital Mic

***HWVAD_IRQn*** Hardware Voice acitivity detect

***USBACT_IRQn*** USB Activity

***USB_IRQn*** USB

***RTC_IRQn*** RTC alarm and wake-up interrupts

***Reserved_IRQn*** Reserved Interrupt

***MAILBOX_IRQn*** Mailbox

***PIN_INT4_IRQn*** External Interrupt 4

***PIN_INT5_IRQn*** External Interrupt 5

***PIN_INT6_IRQn*** External Interrupt 6

***PIN_INT7_IRQn*** External Interrupt 7

***CT32B2_IRQn*** CTMR2

***CT32B4_IRQn*** CTMR4

***Reserved1_IRQn*** Reserved Interrupt

***SPIFI_IRQn*** SPI Flash interface

Definition at line 75 of file cmsis_5411x.h.

## 6.58 CMSIS Core Instruction Interface

Access to dedicated instructions

## 6.59 CMSIS Core Register Access Functions

## 6.60 CMSIS Global Defines

**IO Type Qualifiers** are used

- to specify the access to peripheral variables.
- for automatic generation of peripheral register debug information.

## 6.61 CMSIS SIMD Intrinsics

Access to dedicated SIMD instructions

## 6.62 CMSIS support

ARM CMSIS DSP Library

## 6.63 Chip specific drivers

### 6.63.1 Detailed Description

Chip specific drivers are unique to a specific device. Chip specific drivers may use an IP driver as it's base driver or a custom implementation if that peripheral or IP on the chip is unique (ie, clocking).

#### Modules

- Common components used with chip drivers
- LPC5411x Chip specific drivers

## 6.64 Code Red LPCXpresso support in LPCOpen

LPCXpresso 6

## 6.65 Common FreeRTOS functions shared with multiple platforms

# 6.66 Common components used with chip drivers

## 6.66.1 Detailed Description

### Modules

- CHIP: Common Chip ISP/IAP commands and return codes
- CHIP: FPU initialization
- CHIP: LPC Common Types
- CHIP: RTC tick to (a more) Universal Time conversion functions
- CHIP: Simple ring buffer implementation
- CHIP: Stopwatch primitives.

## 6.67 Community support for LPCOpen

You can also submit questions and comments on the LPCware.com forums at the:
LPCware community forums.

If you find an issue with the LPCOpen code or an example, you can submit bug reports for LPCOpen code at:
NXP Technical support

Note: You will need to create a free LPCware.com account to use the community forums.

## 6.68 Copyright (C) 2013 NXP Semiconductors. All rights reserved.

## 6.69 Core Debug Registers (CoreDebug)

### 6.69.1 Detailed Description

Cortex-M0+ Core Debug Registers (DCB registers, SHCSR, and DFSR) are only accessible over DAP and not via processor. Therefore they are not covered by the Cortex-M0 header file.

Type definitions for the Core Debug Registers.

### Data Structures

- struct CoreDebug_Type

    *Structure type to access the Core Debug Register (CoreDebug).*

### Macros

- #define CoreDebug_DHCSR_DBGKEY_Pos 16
- #define CoreDebug_DHCSR_DBGKEY_Msk (0xFFFFUL << CoreDebug_DHCSR_DBGKEY_Pos)
- #define CoreDebug_DHCSR_S_RESET_ST_Pos 25
- #define CoreDebug_DHCSR_S_RESET_ST_Msk (1UL << CoreDebug_DHCSR_S_RESET_ST_Pos)
- #define CoreDebug_DHCSR_S_RETIRE_ST_Pos 24
- #define CoreDebug_DHCSR_S_RETIRE_ST_Msk (1UL << CoreDebug_DHCSR_S_RETIRE_ST_Pos)
- #define CoreDebug_DHCSR_S_LOCKUP_Pos 19
- #define CoreDebug_DHCSR_S_LOCKUP_Msk (1UL << CoreDebug_DHCSR_S_LOCKUP_Pos)
- #define CoreDebug_DHCSR_S_SLEEP_Pos 18
- #define CoreDebug_DHCSR_S_SLEEP_Msk (1UL << CoreDebug_DHCSR_S_SLEEP_Pos)
- #define CoreDebug_DHCSR_S_HALT_Pos 17
- #define CoreDebug_DHCSR_S_HALT_Msk (1UL << CoreDebug_DHCSR_S_HALT_Pos)
- #define CoreDebug_DHCSR_S_REGRDY_Pos 16
- #define CoreDebug_DHCSR_S_REGRDY_Msk (1UL << CoreDebug_DHCSR_S_REGRDY_Pos)
- #define CoreDebug_DHCSR_C_SNAPSTALL_Pos 5
- #define CoreDebug_DHCSR_C_SNAPSTALL_Msk (1UL << CoreDebug_DHCSR_C_SNAPSTALL_Pos)
- #define CoreDebug_DHCSR_C_MASKINTS_Pos 3
- #define CoreDebug_DHCSR_C_MASKINTS_Msk (1UL << CoreDebug_DHCSR_C_MASKINTS_Pos)
- #define CoreDebug_DHCSR_C_STEP_Pos 2
- #define CoreDebug_DHCSR_C_STEP_Msk (1UL << CoreDebug_DHCSR_C_STEP_Pos)
- #define CoreDebug_DHCSR_C_HALT_Pos 1
- #define CoreDebug_DHCSR_C_HALT_Msk (1UL << CoreDebug_DHCSR_C_HALT_Pos)
- #define CoreDebug_DHCSR_C_DEBUGEN_Pos 0
- #define CoreDebug_DHCSR_C_DEBUGEN_Msk (1UL << CoreDebug_DHCSR_C_DEBUGEN_Pos)
- #define CoreDebug_DCRSR_REGWnR_Pos 16
- #define CoreDebug_DCRSR_REGWnR_Msk (1UL << CoreDebug_DCRSR_REGWnR_Pos)
- #define CoreDebug_DCRSR_REGSEL_Pos 0
- #define CoreDebug_DCRSR_REGSEL_Msk (0x1FUL << CoreDebug_DCRSR_REGSEL_Pos)
- #define CoreDebug_DEMCR_TRCENA_Pos 24
- #define CoreDebug_DEMCR_TRCENA_Msk (1UL << CoreDebug_DEMCR_TRCENA_Pos)
- #define CoreDebug_DEMCR_MON_REQ_Pos 19
- #define CoreDebug_DEMCR_MON_REQ_Msk (1UL << CoreDebug_DEMCR_MON_REQ_Pos)
- #define CoreDebug_DEMCR_MON_STEP_Pos 18
- #define CoreDebug_DEMCR_MON_STEP_Msk (1UL << CoreDebug_DEMCR_MON_STEP_Pos)
- #define CoreDebug_DEMCR_MON_PEND_Pos 17
- #define CoreDebug_DEMCR_MON_PEND_Msk (1UL << CoreDebug_DEMCR_MON_PEND_Pos)
- #define CoreDebug_DEMCR_MON_EN_Pos 16
- #define CoreDebug_DEMCR_MON_EN_Msk (1UL << CoreDebug_DEMCR_MON_EN_Pos)

- #define CoreDebug_DEMCR_VC_HARDERR_Pos 10
- #define CoreDebug_DEMCR_VC_HARDERR_Msk (1UL << CoreDebug_DEMCR_VC_HARDERR_Pos)
- #define CoreDebug_DEMCR_VC_INTERR_Pos 9
- #define CoreDebug_DEMCR_VC_INTERR_Msk (1UL << CoreDebug_DEMCR_VC_INTERR_Pos)
- #define CoreDebug_DEMCR_VC_BUSERR_Pos 8
- #define CoreDebug_DEMCR_VC_BUSERR_Msk (1UL << CoreDebug_DEMCR_VC_BUSERR_Pos)
- #define CoreDebug_DEMCR_VC_STATERR_Pos 7
- #define CoreDebug_DEMCR_VC_STATERR_Msk (1UL << CoreDebug_DEMCR_VC_STATERR_Pos)
- #define CoreDebug_DEMCR_VC_CHKERR_Pos 6
- #define CoreDebug_DEMCR_VC_CHKERR_Msk (1UL << CoreDebug_DEMCR_VC_CHKERR_Pos)
- #define CoreDebug_DEMCR_VC_NOCPERR_Pos 5
- #define CoreDebug_DEMCR_VC_NOCPERR_Msk (1UL << CoreDebug_DEMCR_VC_NOCPERR_Pos)
- #define CoreDebug_DEMCR_VC_MMERR_Pos 4
- #define CoreDebug_DEMCR_VC_MMERR_Msk (1UL << CoreDebug_DEMCR_VC_MMERR_Pos)
- #define CoreDebug_DEMCR_VC_CORERESET_Pos 0
- #define CoreDebug_DEMCR_VC_CORERESET_Msk (1UL << CoreDebug_DEMCR_VC_CORERESET↩_Pos)

## 6.69.2 Macro Definition Documentation

### 6.69.2.1 #define CoreDebug_DCRSR_REGSEL_Msk (0x1FUL << CoreDebug_DCRSR_REGSEL_Pos)

CoreDebug DCRSR: REGSEL Mask

Definition at line 1321 of file core_cm4.h.

### 6.69.2.2 #define CoreDebug_DCRSR_REGSEL_Pos 0

CoreDebug DCRSR: REGSEL Position

Definition at line 1320 of file core_cm4.h.

### 6.69.2.3 #define CoreDebug_DCRSR_REGWnR_Msk (1UL << CoreDebug_DCRSR_REGWnR_Pos)

CoreDebug DCRSR: REGWnR Mask

Definition at line 1318 of file core_cm4.h.

### 6.69.2.4 #define CoreDebug_DCRSR_REGWnR_Pos 16

CoreDebug DCRSR: REGWnR Position

Definition at line 1317 of file core_cm4.h.

### 6.69.2.5 #define CoreDebug_DEMCR_MON_EN_Msk (1UL << CoreDebug_DEMCR_MON_EN_Pos)

CoreDebug DEMCR: MON_EN Mask

Definition at line 1337 of file core_cm4.h.

### 6.69.2.6 #define CoreDebug_DEMCR_MON_EN_Pos 16

CoreDebug DEMCR: MON_EN Position

Definition at line 1336 of file core_cm4.h.

#### 6.69.2.7 #define CoreDebug_DEMCR_MON_PEND_Msk (1UL << CoreDebug_DEMCR_MON_PEND_Pos)

CoreDebug DEMCR: MON_PEND Mask

Definition at line 1334 of file core_cm4.h.

#### 6.69.2.8 #define CoreDebug_DEMCR_MON_PEND_Pos 17

CoreDebug DEMCR: MON_PEND Position

Definition at line 1333 of file core_cm4.h.

#### 6.69.2.9 #define CoreDebug_DEMCR_MON_REQ_Msk (1UL << CoreDebug_DEMCR_MON_REQ_Pos)

CoreDebug DEMCR: MON_REQ Mask

Definition at line 1328 of file core_cm4.h.

#### 6.69.2.10 #define CoreDebug_DEMCR_MON_REQ_Pos 19

CoreDebug DEMCR: MON_REQ Position

Definition at line 1327 of file core_cm4.h.

#### 6.69.2.11 #define CoreDebug_DEMCR_MON_STEP_Msk (1UL << CoreDebug_DEMCR_MON_STEP_Pos)

CoreDebug DEMCR: MON_STEP Mask

Definition at line 1331 of file core_cm4.h.

#### 6.69.2.12 #define CoreDebug_DEMCR_MON_STEP_Pos 18

CoreDebug DEMCR: MON_STEP Position

Definition at line 1330 of file core_cm4.h.

#### 6.69.2.13 #define CoreDebug_DEMCR_TRCENA_Msk (1UL << CoreDebug_DEMCR_TRCENA_Pos)

CoreDebug DEMCR: TRCENA Mask

Definition at line 1325 of file core_cm4.h.

#### 6.69.2.14 #define CoreDebug_DEMCR_TRCENA_Pos 24

CoreDebug DEMCR: TRCENA Position

Definition at line 1324 of file core_cm4.h.

#### 6.69.2.15 #define CoreDebug_DEMCR_VC_BUSERR_Msk (1UL << CoreDebug_DEMCR_VC_BUSERR_Pos)

CoreDebug DEMCR: VC_BUSERR Mask

Definition at line 1346 of file core_cm4.h.

#### 6.69.2.16 #define CoreDebug_DEMCR_VC_BUSERR_Pos 8

CoreDebug DEMCR: VC_BUSERR Position

Definition at line 1345 of file core_cm4.h.

#### 6.69.2.17 #define CoreDebug_DEMCR_VC_CHKERR_Msk (1UL << CoreDebug_DEMCR_VC_CHKERR_Pos)

CoreDebug DEMCR: VC_CHKERR Mask

Definition at line 1352 of file core_cm4.h.

#### 6.69.2.18 #define CoreDebug_DEMCR_VC_CHKERR_Pos 6

CoreDebug DEMCR: VC_CHKERR Position

Definition at line 1351 of file core_cm4.h.

#### 6.69.2.19 #define CoreDebug_DEMCR_VC_CORERESET_Msk (1UL << CoreDebug_DEMCR_VC_CORERESET_Pos)

CoreDebug DEMCR: VC_CORERESET Mask

Definition at line 1361 of file core_cm4.h.

#### 6.69.2.20 #define CoreDebug_DEMCR_VC_CORERESET_Pos 0

CoreDebug DEMCR: VC_CORERESET Position

Definition at line 1360 of file core_cm4.h.

#### 6.69.2.21 #define CoreDebug_DEMCR_VC_HARDERR_Msk (1UL << CoreDebug_DEMCR_VC_HARDERR_Pos)

CoreDebug DEMCR: VC_HARDERR Mask

Definition at line 1340 of file core_cm4.h.

#### 6.69.2.22 #define CoreDebug_DEMCR_VC_HARDERR_Pos 10

CoreDebug DEMCR: VC_HARDERR Position

Definition at line 1339 of file core_cm4.h.

#### 6.69.2.23 #define CoreDebug_DEMCR_VC_INTERR_Msk (1UL << CoreDebug_DEMCR_VC_INTERR_Pos)

CoreDebug DEMCR: VC_INTERR Mask

Definition at line 1343 of file core_cm4.h.

#### 6.69.2.24 #define CoreDebug_DEMCR_VC_INTERR_Pos 9

CoreDebug DEMCR: VC_INTERR Position

Definition at line 1342 of file core_cm4.h.

#### 6.69.2.25 #define CoreDebug_DEMCR_VC_MMERR_Msk (1UL << CoreDebug_DEMCR_VC_MMERR_Pos)

CoreDebug DEMCR: VC_MMERR Mask

Definition at line 1358 of file core_cm4.h.

#### 6.69.2.26 #define CoreDebug_DEMCR_VC_MMERR_Pos 4

CoreDebug DEMCR: VC_MMERR Position

Definition at line 1357 of file core_cm4.h.

#### 6.69.2.27 #define CoreDebug_DEMCR_VC_NOCPERR_Msk (1UL << CoreDebug_DEMCR_VC_NOCPERR_Pos)

CoreDebug DEMCR: VC_NOCPERR Mask

Definition at line 1355 of file core_cm4.h.

#### 6.69.2.28 #define CoreDebug_DEMCR_VC_NOCPERR_Pos 5

CoreDebug DEMCR: VC_NOCPERR Position

Definition at line 1354 of file core_cm4.h.

#### 6.69.2.29 #define CoreDebug_DEMCR_VC_STATERR_Msk (1UL << CoreDebug_DEMCR_VC_STATERR_Pos)

CoreDebug DEMCR: VC_STATERR Mask

Definition at line 1349 of file core_cm4.h.

#### 6.69.2.30 #define CoreDebug_DEMCR_VC_STATERR_Pos 7

CoreDebug DEMCR: VC_STATERR Position

Definition at line 1348 of file core_cm4.h.

#### 6.69.2.31 #define CoreDebug_DHCSR_C_DEBUGEN_Msk (1UL << CoreDebug_DHCSR_C_DEBUGEN_Pos)

CoreDebug DHCSR: C_DEBUGEN Mask

Definition at line 1314 of file core_cm4.h.

#### 6.69.2.32 #define CoreDebug_DHCSR_C_DEBUGEN_Pos 0

CoreDebug DHCSR: C_DEBUGEN Position

Definition at line 1313 of file core_cm4.h.

#### 6.69.2.33 #define CoreDebug_DHCSR_C_HALT_Msk (1UL << CoreDebug_DHCSR_C_HALT_Pos)

CoreDebug DHCSR: C_HALT Mask

Definition at line 1311 of file core_cm4.h.

#### 6.69.2.34 #define CoreDebug_DHCSR_C_HALT_Pos 1

CoreDebug DHCSR: C_HALT Position

Definition at line 1310 of file core_cm4.h.

#### 6.69.2.35 #define CoreDebug_DHCSR_C_MASKINTS_Msk (1UL << CoreDebug_DHCSR_C_MASKINTS_Pos)

CoreDebug DHCSR: C_MASKINTS Mask

Definition at line 1305 of file core_cm4.h.

#### 6.69.2.36 #define CoreDebug_DHCSR_C_MASKINTS_Pos 3

CoreDebug DHCSR: C_MASKINTS Position

Definition at line 1304 of file core_cm4.h.

#### 6.69.2.37 #define CoreDebug_DHCSR_C_SNAPSTALL_Msk (1UL << CoreDebug_DHCSR_C_SNAPSTALL_Pos)

CoreDebug DHCSR: C_SNAPSTALL Mask

Definition at line 1302 of file core_cm4.h.

#### 6.69.2.38 #define CoreDebug_DHCSR_C_SNAPSTALL_Pos 5

CoreDebug DHCSR: C_SNAPSTALL Position

Definition at line 1301 of file core_cm4.h.

#### 6.69.2.39 #define CoreDebug_DHCSR_C_STEP_Msk (1UL << CoreDebug_DHCSR_C_STEP_Pos)

CoreDebug DHCSR: C_STEP Mask

Definition at line 1308 of file core_cm4.h.

#### 6.69.2.40 #define CoreDebug_DHCSR_C_STEP_Pos 2

CoreDebug DHCSR: C_STEP Position

Definition at line 1307 of file core_cm4.h.

#### 6.69.2.41 #define CoreDebug_DHCSR_DBGKEY_Msk (0xFFFFUL << CoreDebug_DHCSR_DBGKEY_Pos)

CoreDebug DHCSR: DBGKEY Mask

Definition at line 1281 of file core_cm4.h.

#### 6.69.2.42 #define CoreDebug_DHCSR_DBGKEY_Pos 16

CoreDebug DHCSR: DBGKEY Position

Definition at line 1280 of file core_cm4.h.

#### 6.69.2.43 #define CoreDebug_DHCSR_S_HALT_Msk (1UL << CoreDebug_DHCSR_S_HALT_Pos)

CoreDebug DHCSR: S_HALT Mask

Definition at line 1296 of file core_cm4.h.

#### 6.69.2.44 #define CoreDebug_DHCSR_S_HALT_Pos 17

CoreDebug DHCSR: S_HALT Position

Definition at line 1295 of file core_cm4.h.

#### 6.69.2.45 #define CoreDebug_DHCSR_S_LOCKUP_Msk (1UL << CoreDebug_DHCSR_S_LOCKUP_Pos)

CoreDebug DHCSR: S_LOCKUP Mask

Definition at line 1290 of file core_cm4.h.

#### 6.69.2.46 #define CoreDebug_DHCSR_S_LOCKUP_Pos 19

CoreDebug DHCSR: S_LOCKUP Position

Definition at line 1289 of file core_cm4.h.

#### 6.69.2.47 #define CoreDebug_DHCSR_S_REGRDY_Msk (1UL << CoreDebug_DHCSR_S_REGRDY_Pos)

CoreDebug DHCSR: S_REGRDY Mask

Definition at line 1299 of file core_cm4.h.

#### 6.69.2.48 #define CoreDebug_DHCSR_S_REGRDY_Pos 16

CoreDebug DHCSR: S_REGRDY Position

Definition at line 1298 of file core_cm4.h.

#### 6.69.2.49 #define CoreDebug_DHCSR_S_RESET_ST_Msk (1UL << CoreDebug_DHCSR_S_RESET_ST_Pos)

CoreDebug DHCSR: S_RESET_ST Mask

Definition at line 1284 of file core_cm4.h.

#### 6.69.2.50 #define CoreDebug_DHCSR_S_RESET_ST_Pos 25

CoreDebug DHCSR: S_RESET_ST Position

Definition at line 1283 of file core_cm4.h.

#### 6.69.2.51 #define CoreDebug_DHCSR_S_RETIRE_ST_Msk (1UL << CoreDebug_DHCSR_S_RETIRE_ST_Pos)

CoreDebug DHCSR: S_RETIRE_ST Mask

Definition at line 1287 of file core_cm4.h.

#### 6.69.2.52 #define CoreDebug_DHCSR_S_RETIRE_ST_Pos 24

CoreDebug DHCSR: S_RETIRE_ST Position

Definition at line 1286 of file core_cm4.h.

#### 6.69.2.53 #define CoreDebug_DHCSR_S_SLEEP_Msk (1UL << CoreDebug_DHCSR_S_SLEEP_Pos)

CoreDebug DHCSR: S_SLEEP Mask

Definition at line 1293 of file core_cm4.h.

#### 6.69.2.54 #define CoreDebug_DHCSR_S_SLEEP_Pos 18

CoreDebug DHCSR: S_SLEEP Position

Definition at line 1292 of file core_cm4.h.

# 6.70 Core Definitions

## 6.70.1 Detailed Description

Definitions for base addresses, unions, and structures.

### Macros

- #define SCS_BASE (0xE000E000UL)
- #define SysTick_BASE (SCS_BASE + 0x0010UL)
- #define NVIC_BASE (SCS_BASE + 0x0100UL)
- #define SCB_BASE (SCS_BASE + 0x0D00UL)
- #define SCB ((SCB_Type ∗) SCB_BASE )
- #define SysTick ((SysTick_Type ∗) SysTick_BASE )
- #define NVIC ((NVIC_Type ∗) NVIC_BASE )
- #define SCS_BASE (0xE000E000UL)
- #define ITM_BASE (0xE0000000UL)
- #define DWT_BASE (0xE0001000UL)
- #define TPI_BASE (0xE0040000UL)
- #define CoreDebug_BASE (0xE000EDF0UL)
- #define SysTick_BASE (SCS_BASE + 0x0010UL)
- #define NVIC_BASE (SCS_BASE + 0x0100UL)
- #define SCB_BASE (SCS_BASE + 0x0D00UL)
- #define SCnSCB ((SCnSCB_Type ∗) SCS_BASE )
- #define SCB ((SCB_Type ∗) SCB_BASE )
- #define SysTick ((SysTick_Type ∗) SysTick_BASE )
- #define NVIC ((NVIC_Type ∗) NVIC_BASE )
- #define ITM ((ITM_Type ∗) ITM_BASE )
- #define DWT ((DWT_Type ∗) DWT_BASE )
- #define TPI ((TPI_Type ∗) TPI_BASE )
- #define CoreDebug ((CoreDebug_Type ∗) CoreDebug_BASE)

## 6.70.2 Macro Definition Documentation

### 6.70.2.1 #define CoreDebug ((CoreDebug_Type ∗) CoreDebug_BASE)

Core Debug configuration struct

Definition at line 1389 of file core_cm4.h.

### 6.70.2.2 #define CoreDebug_BASE (0xE000EDF0UL)

Core Debug Base Address

Definition at line 1377 of file core_cm4.h.

### 6.70.2.3 #define DWT ((DWT_Type ∗) DWT_BASE )

DWT configuration struct

Definition at line 1387 of file core_cm4.h.

#### 6.70.2.4 #define DWT_BASE (0xE0001000UL)

DWT Base Address

Definition at line 1375 of file core_cm4.h.

#### 6.70.2.5 #define ITM ((ITM_Type ∗) ITM_BASE )

ITM configuration struct

Definition at line 1386 of file core_cm4.h.

#### 6.70.2.6 #define ITM_BASE (0xE0000000UL)

ITM Base Address

Definition at line 1374 of file core_cm4.h.

#### 6.70.2.7 #define NVIC ((NVIC_Type ∗) NVIC_BASE )

NVIC configuration struct

Definition at line 583 of file core_cm0plus.h.

#### 6.70.2.8 #define NVIC ((NVIC_Type ∗) NVIC_BASE )

NVIC configuration struct

Definition at line 1385 of file core_cm4.h.

#### 6.70.2.9 #define NVIC_BASE (SCS_BASE + 0x0100UL)

NVIC Base Address

Definition at line 578 of file core_cm0plus.h.

#### 6.70.2.10 #define NVIC_BASE (SCS_BASE + 0x0100UL)

NVIC Base Address

Definition at line 1379 of file core_cm4.h.

#### 6.70.2.11 #define SCB ((SCB_Type ∗) SCB_BASE )

SCB configuration struct

Definition at line 581 of file core_cm0plus.h.

#### 6.70.2.12 #define SCB ((SCB_Type ∗) SCB_BASE )

SCB configuration struct

Definition at line 1383 of file core_cm4.h.

#### 6.70.2.13 #define SCB_BASE (SCS_BASE + 0x0D00UL)

System Control Block Base Address

Definition at line 579 of file core_cm0plus.h.

#### 6.70.2.14 #define SCB_BASE (SCS_BASE + 0x0D00UL)

System Control Block Base Address

Definition at line 1380 of file core_cm4.h.

#### 6.70.2.15 #define SCnSCB ((SCnSCB_Type *) SCS_BASE )

System control Register not in SCB

Definition at line 1382 of file core_cm4.h.

#### 6.70.2.16 #define SCS_BASE (0xE000E000UL)

System Control Space Base Address

Definition at line 576 of file core_cm0plus.h.

#### 6.70.2.17 #define SCS_BASE (0xE000E000UL)

System Control Space Base Address

Definition at line 1373 of file core_cm4.h.

#### 6.70.2.18 #define SysTick ((SysTick_Type *) SysTick_BASE )

SysTick configuration struct

Definition at line 582 of file core_cm0plus.h.

#### 6.70.2.19 #define SysTick ((SysTick_Type *) SysTick_BASE )

SysTick configuration struct

Definition at line 1384 of file core_cm4.h.

#### 6.70.2.20 #define SysTick_BASE (SCS_BASE + 0x0010UL)

SysTick Base Address

Definition at line 577 of file core_cm0plus.h.

#### 6.70.2.21 #define SysTick_BASE (SCS_BASE + 0x0010UL)

SysTick Base Address

Definition at line 1378 of file core_cm4.h.

#### 6.70.2.22 #define TPI ((TPI_Type ∗) TPI_BASE )

TPI configuration struct

Definition at line 1388 of file core_cm4.h.

#### 6.70.2.23 #define TPI_BASE (0xE0040000UL)

TPI Base Address

Definition at line 1376 of file core_cm4.h.

## 6.71 Data Watchpoint and Trace (DWT)

### 6.71.1 Detailed Description

Type definitions for the Data Watchpoint and Trace (DWT)

### Data Structures

- struct DWT_Type

  *Structure type to access the Data Watchpoint and Trace Register (DWT).*

### Macros

- #define DWT_CTRL_NUMCOMP_Pos 28
- #define DWT_CTRL_NUMCOMP_Msk (0xFUL << DWT_CTRL_NUMCOMP_Pos)
- #define DWT_CTRL_NOTRCPKT_Pos 27
- #define DWT_CTRL_NOTRCPKT_Msk (0x1UL << DWT_CTRL_NOTRCPKT_Pos)
- #define DWT_CTRL_NOEXTTRIG_Pos 26
- #define DWT_CTRL_NOEXTTRIG_Msk (0x1UL << DWT_CTRL_NOEXTTRIG_Pos)
- #define DWT_CTRL_NOCYCCNT_Pos 25
- #define DWT_CTRL_NOCYCCNT_Msk (0x1UL << DWT_CTRL_NOCYCCNT_Pos)
- #define DWT_CTRL_NOPRFCNT_Pos 24
- #define DWT_CTRL_NOPRFCNT_Msk (0x1UL << DWT_CTRL_NOPRFCNT_Pos)
- #define DWT_CTRL_CYCEVTENA_Pos 22
- #define DWT_CTRL_CYCEVTENA_Msk (0x1UL << DWT_CTRL_CYCEVTENA_Pos)
- #define DWT_CTRL_FOLDEVTENA_Pos 21
- #define DWT_CTRL_FOLDEVTENA_Msk (0x1UL << DWT_CTRL_FOLDEVTENA_Pos)
- #define DWT_CTRL_LSUEVTENA_Pos 20
- #define DWT_CTRL_LSUEVTENA_Msk (0x1UL << DWT_CTRL_LSUEVTENA_Pos)
- #define DWT_CTRL_SLEEPEVTENA_Pos 19
- #define DWT_CTRL_SLEEPEVTENA_Msk (0x1UL << DWT_CTRL_SLEEPEVTENA_Pos)
- #define DWT_CTRL_EXCEVTENA_Pos 18
- #define DWT_CTRL_EXCEVTENA_Msk (0x1UL << DWT_CTRL_EXCEVTENA_Pos)
- #define DWT_CTRL_CPIEVTENA_Pos 17
- #define DWT_CTRL_CPIEVTENA_Msk (0x1UL << DWT_CTRL_CPIEVTENA_Pos)
- #define DWT_CTRL_EXCTRCENA_Pos 16
- #define DWT_CTRL_EXCTRCENA_Msk (0x1UL << DWT_CTRL_EXCTRCENA_Pos)
- #define DWT_CTRL_PCSAMPLENA_Pos 12
- #define DWT_CTRL_PCSAMPLENA_Msk (0x1UL << DWT_CTRL_PCSAMPLENA_Pos)
- #define DWT_CTRL_SYNCTAP_Pos 10
- #define DWT_CTRL_SYNCTAP_Msk (0x3UL << DWT_CTRL_SYNCTAP_Pos)
- #define DWT_CTRL_CYCTAP_Pos 9
- #define DWT_CTRL_CYCTAP_Msk (0x1UL << DWT_CTRL_CYCTAP_Pos)
- #define DWT_CTRL_POSTINIT_Pos 5
- #define DWT_CTRL_POSTINIT_Msk (0xFUL << DWT_CTRL_POSTINIT_Pos)
- #define DWT_CTRL_POSTPRESET_Pos 1
- #define DWT_CTRL_POSTPRESET_Msk (0xFUL << DWT_CTRL_POSTPRESET_Pos)
- #define DWT_CTRL_CYCCNTENA_Pos 0
- #define DWT_CTRL_CYCCNTENA_Msk (0x1UL << DWT_CTRL_CYCCNTENA_Pos)
- #define DWT_CPICNT_CPICNT_Pos 0
- #define DWT_CPICNT_CPICNT_Msk (0xFFUL << DWT_CPICNT_CPICNT_Pos)
- #define DWT_EXCCNT_EXCCNT_Pos 0
- #define DWT_EXCCNT_EXCCNT_Msk (0xFFUL << DWT_EXCCNT_EXCCNT_Pos)

- #define DWT_SLEEPCNT_SLEEPCNT_Pos 0
- #define DWT_SLEEPCNT_SLEEPCNT_Msk (0xFFUL << DWT_SLEEPCNT_SLEEPCNT_Pos)
- #define DWT_LSUCNT_LSUCNT_Pos 0
- #define DWT_LSUCNT_LSUCNT_Msk (0xFFUL << DWT_LSUCNT_LSUCNT_Pos)
- #define DWT_FOLDCNT_FOLDCNT_Pos 0
- #define DWT_FOLDCNT_FOLDCNT_Msk (0xFFUL << DWT_FOLDCNT_FOLDCNT_Pos)
- #define DWT_MASK_MASK_Pos 0
- #define DWT_MASK_MASK_Msk (0x1FUL << DWT_MASK_MASK_Pos)
- #define DWT_FUNCTION_MATCHED_Pos 24
- #define DWT_FUNCTION_MATCHED_Msk (0x1UL << DWT_FUNCTION_MATCHED_Pos)
- #define DWT_FUNCTION_DATAVADDR1_Pos 16
- #define DWT_FUNCTION_DATAVADDR1_Msk (0xFUL << DWT_FUNCTION_DATAVADDR1_Pos)
- #define DWT_FUNCTION_DATAVADDR0_Pos 12
- #define DWT_FUNCTION_DATAVADDR0_Msk (0xFUL << DWT_FUNCTION_DATAVADDR0_Pos)
- #define DWT_FUNCTION_DATAVSIZE_Pos 10
- #define DWT_FUNCTION_DATAVSIZE_Msk (0x3UL << DWT_FUNCTION_DATAVSIZE_Pos)
- #define DWT_FUNCTION_LNK1ENA_Pos 9
- #define DWT_FUNCTION_LNK1ENA_Msk (0x1UL << DWT_FUNCTION_LNK1ENA_Pos)
- #define DWT_FUNCTION_DATAVMATCH_Pos 8
- #define DWT_FUNCTION_DATAVMATCH_Msk (0x1UL << DWT_FUNCTION_DATAVMATCH_Pos)
- #define DWT_FUNCTION_CYCMATCH_Pos 7
- #define DWT_FUNCTION_CYCMATCH_Msk (0x1UL << DWT_FUNCTION_CYCMATCH_Pos)
- #define DWT_FUNCTION_EMITRANGE_Pos 5
- #define DWT_FUNCTION_EMITRANGE_Msk (0x1UL << DWT_FUNCTION_EMITRANGE_Pos)
- #define DWT_FUNCTION_FUNCTION_Pos 0
- #define DWT_FUNCTION_FUNCTION_Msk (0xFUL << DWT_FUNCTION_FUNCTION_Pos)

## 6.71.2 Macro Definition Documentation

### 6.71.2.1 #define DWT_CPICNT_CPICNT_Msk (0xFFUL << DWT_CPICNT_CPICNT_Pos)

DWT CPICNT: CPICNT Mask

Definition at line 858 of file core_cm4.h.

### 6.71.2.2 #define DWT_CPICNT_CPICNT_Pos 0

DWT CPICNT: CPICNT Position

Definition at line 857 of file core_cm4.h.

### 6.71.2.3 #define DWT_CTRL_CPIEVTENA_Msk (0x1UL << DWT_CTRL_CPIEVTENA_Pos)

DWT CTRL: CPIEVTENA Mask

Definition at line 833 of file core_cm4.h.

### 6.71.2.4 #define DWT_CTRL_CPIEVTENA_Pos 17

DWT CTRL: CPIEVTENA Position

Definition at line 832 of file core_cm4.h.

#### 6.71.2.5 #define DWT_CTRL_CYCCNTENA_Msk (0x1UL << DWT_CTRL_CYCCNTENA_Pos)

DWT CTRL: CYCCNTENA Mask

Definition at line 854 of file core_cm4.h.

#### 6.71.2.6 #define DWT_CTRL_CYCCNTENA_Pos 0

DWT CTRL: CYCCNTENA Position

Definition at line 853 of file core_cm4.h.

#### 6.71.2.7 #define DWT_CTRL_CYCEVTENA_Msk (0x1UL << DWT_CTRL_CYCEVTENA_Pos)

DWT CTRL: CYCEVTENA Mask

Definition at line 818 of file core_cm4.h.

#### 6.71.2.8 #define DWT_CTRL_CYCEVTENA_Pos 22

DWT CTRL: CYCEVTENA Position

Definition at line 817 of file core_cm4.h.

#### 6.71.2.9 #define DWT_CTRL_CYCTAP_Msk (0x1UL << DWT_CTRL_CYCTAP_Pos)

DWT CTRL: CYCTAP Mask

Definition at line 845 of file core_cm4.h.

#### 6.71.2.10 #define DWT_CTRL_CYCTAP_Pos 9

DWT CTRL: CYCTAP Position

Definition at line 844 of file core_cm4.h.

#### 6.71.2.11 #define DWT_CTRL_EXCEVTENA_Msk (0x1UL << DWT_CTRL_EXCEVTENA_Pos)

DWT CTRL: EXCEVTENA Mask

Definition at line 830 of file core_cm4.h.

#### 6.71.2.12 #define DWT_CTRL_EXCEVTENA_Pos 18

DWT CTRL: EXCEVTENA Position

Definition at line 829 of file core_cm4.h.

#### 6.71.2.13 #define DWT_CTRL_EXCTRCENA_Msk (0x1UL << DWT_CTRL_EXCTRCENA_Pos)

DWT CTRL: EXCTRCENA Mask

Definition at line 836 of file core_cm4.h.

#### 6.71.2.14 #define DWT_CTRL_EXCTRCENA_Pos 16

DWT CTRL: EXCTRCENA Position

Definition at line 835 of file core_cm4.h.

#### 6.71.2.15 #define DWT_CTRL_FOLDEVTENA_Msk (0x1UL << DWT_CTRL_FOLDEVTENA_Pos)

DWT CTRL: FOLDEVTENA Mask

Definition at line 821 of file core_cm4.h.

#### 6.71.2.16 #define DWT_CTRL_FOLDEVTENA_Pos 21

DWT CTRL: FOLDEVTENA Position

Definition at line 820 of file core_cm4.h.

#### 6.71.2.17 #define DWT_CTRL_LSUEVTENA_Msk (0x1UL << DWT_CTRL_LSUEVTENA_Pos)

DWT CTRL: LSUEVTENA Mask

Definition at line 824 of file core_cm4.h.

#### 6.71.2.18 #define DWT_CTRL_LSUEVTENA_Pos 20

DWT CTRL: LSUEVTENA Position

Definition at line 823 of file core_cm4.h.

#### 6.71.2.19 #define DWT_CTRL_NOCYCCNT_Msk (0x1UL << DWT_CTRL_NOCYCCNT_Pos)

DWT CTRL: NOCYCCNT Mask

Definition at line 812 of file core_cm4.h.

#### 6.71.2.20 #define DWT_CTRL_NOCYCCNT_Pos 25

DWT CTRL: NOCYCCNT Position

Definition at line 811 of file core_cm4.h.

#### 6.71.2.21 #define DWT_CTRL_NOEXTTRIG_Msk (0x1UL << DWT_CTRL_NOEXTTRIG_Pos)

DWT CTRL: NOEXTTRIG Mask

Definition at line 809 of file core_cm4.h.

#### 6.71.2.22 #define DWT_CTRL_NOEXTTRIG_Pos 26

DWT CTRL: NOEXTTRIG Position

Definition at line 808 of file core_cm4.h.

#### 6.71.2.23 #define DWT_CTRL_NOPRFCNT_Msk (0x1UL << DWT_CTRL_NOPRFCNT_Pos)

DWT CTRL: NOPRFCNT Mask

Definition at line 815 of file core_cm4.h.

#### 6.71.2.24 #define DWT_CTRL_NOPRFCNT_Pos 24

DWT CTRL: NOPRFCNT Position

Definition at line 814 of file core_cm4.h.

#### 6.71.2.25 #define DWT_CTRL_NOTRCPKT_Msk (0x1UL << DWT_CTRL_NOTRCPKT_Pos)

DWT CTRL: NOTRCPKT Mask

Definition at line 806 of file core_cm4.h.

#### 6.71.2.26 #define DWT_CTRL_NOTRCPKT_Pos 27

DWT CTRL: NOTRCPKT Position

Definition at line 805 of file core_cm4.h.

#### 6.71.2.27 #define DWT_CTRL_NUMCOMP_Msk (0xFUL << DWT_CTRL_NUMCOMP_Pos)

DWT CTRL: NUMCOMP Mask

Definition at line 803 of file core_cm4.h.

#### 6.71.2.28 #define DWT_CTRL_NUMCOMP_Pos 28

DWT CTRL: NUMCOMP Position

Definition at line 802 of file core_cm4.h.

#### 6.71.2.29 #define DWT_CTRL_PCSAMPLENA_Msk (0x1UL << DWT_CTRL_PCSAMPLENA_Pos)

DWT CTRL: PCSAMPLENA Mask

Definition at line 839 of file core_cm4.h.

#### 6.71.2.30 #define DWT_CTRL_PCSAMPLENA_Pos 12

DWT CTRL: PCSAMPLENA Position

Definition at line 838 of file core_cm4.h.

#### 6.71.2.31 #define DWT_CTRL_POSTINIT_Msk (0xFUL << DWT_CTRL_POSTINIT_Pos)

DWT CTRL: POSTINIT Mask

Definition at line 848 of file core_cm4.h.

#### 6.71.2.32 #define DWT_CTRL_POSTINIT_Pos 5

DWT CTRL: POSTINIT Position

Definition at line 847 of file core_cm4.h.

#### 6.71.2.33 #define DWT_CTRL_POSTPRESET_Msk (0xFUL << DWT_CTRL_POSTPRESET_Pos)

DWT CTRL: POSTPRESET Mask

Definition at line 851 of file core_cm4.h.

#### 6.71.2.34 #define DWT_CTRL_POSTPRESET_Pos 1

DWT CTRL: POSTPRESET Position

Definition at line 850 of file core_cm4.h.

#### 6.71.2.35 #define DWT_CTRL_SLEEPEVTENA_Msk (0x1UL << DWT_CTRL_SLEEPEVTENA_Pos)

DWT CTRL: SLEEPEVTENA Mask

Definition at line 827 of file core_cm4.h.

#### 6.71.2.36 #define DWT_CTRL_SLEEPEVTENA_Pos 19

DWT CTRL: SLEEPEVTENA Position

Definition at line 826 of file core_cm4.h.

#### 6.71.2.37 #define DWT_CTRL_SYNCTAP_Msk (0x3UL << DWT_CTRL_SYNCTAP_Pos)

DWT CTRL: SYNCTAP Mask

Definition at line 842 of file core_cm4.h.

#### 6.71.2.38 #define DWT_CTRL_SYNCTAP_Pos 10

DWT CTRL: SYNCTAP Position

Definition at line 841 of file core_cm4.h.

#### 6.71.2.39 #define DWT_EXCCNT_EXCCNT_Msk (0xFFUL << DWT_EXCCNT_EXCCNT_Pos)

DWT EXCCNT: EXCCNT Mask

Definition at line 862 of file core_cm4.h.

#### 6.71.2.40 #define DWT_EXCCNT_EXCCNT_Pos 0

DWT EXCCNT: EXCCNT Position

Definition at line 861 of file core_cm4.h.

#### 6.71.2.41 #define DWT_FOLDCNT_FOLDCNT_Msk (0xFFUL << DWT_FOLDCNT_FOLDCNT_Pos)

DWT FOLDCNT: FOLDCNT Mask

Definition at line 874 of file core_cm4.h.

#### 6.71.2.42 #define DWT_FOLDCNT_FOLDCNT_Pos 0

DWT FOLDCNT: FOLDCNT Position

Definition at line 873 of file core_cm4.h.

#### 6.71.2.43 #define DWT_FUNCTION_CYCMATCH_Msk (0x1UL << DWT_FUNCTION_CYCMATCH_Pos)

DWT FUNCTION: CYCMATCH Mask

Definition at line 900 of file core_cm4.h.

#### 6.71.2.44 #define DWT_FUNCTION_CYCMATCH_Pos 7

DWT FUNCTION: CYCMATCH Position

Definition at line 899 of file core_cm4.h.

#### 6.71.2.45 #define DWT_FUNCTION_DATAVADDR0_Msk (0xFUL << DWT_FUNCTION_DATAVADDR0_Pos)

DWT FUNCTION: DATAVADDR0 Mask

Definition at line 888 of file core_cm4.h.

#### 6.71.2.46 #define DWT_FUNCTION_DATAVADDR0_Pos 12

DWT FUNCTION: DATAVADDR0 Position

Definition at line 887 of file core_cm4.h.

#### 6.71.2.47 #define DWT_FUNCTION_DATAVADDR1_Msk (0xFUL << DWT_FUNCTION_DATAVADDR1_Pos)

DWT FUNCTION: DATAVADDR1 Mask

Definition at line 885 of file core_cm4.h.

#### 6.71.2.48 #define DWT_FUNCTION_DATAVADDR1_Pos 16

DWT FUNCTION: DATAVADDR1 Position

Definition at line 884 of file core_cm4.h.

#### 6.71.2.49 #define DWT_FUNCTION_DATAVMATCH_Msk (0x1UL << DWT_FUNCTION_DATAVMATCH_Pos)

DWT FUNCTION: DATAVMATCH Mask

Definition at line 897 of file core_cm4.h.

#### 6.71.2.50 #define DWT_FUNCTION_DATAVMATCH_Pos 8

DWT FUNCTION: DATAVMATCH Position

Definition at line 896 of file core_cm4.h.

#### 6.71.2.51 #define DWT_FUNCTION_DATAVSIZE_Msk (0x3UL << DWT_FUNCTION_DATAVSIZE_Pos)

DWT FUNCTION: DATAVSIZE Mask

Definition at line 891 of file core_cm4.h.

#### 6.71.2.52 #define DWT_FUNCTION_DATAVSIZE_Pos 10

DWT FUNCTION: DATAVSIZE Position

Definition at line 890 of file core_cm4.h.

#### 6.71.2.53 #define DWT_FUNCTION_EMITRANGE_Msk (0x1UL << DWT_FUNCTION_EMITRANGE_Pos)

DWT FUNCTION: EMITRANGE Mask

Definition at line 903 of file core_cm4.h.

#### 6.71.2.54 #define DWT_FUNCTION_EMITRANGE_Pos 5

DWT FUNCTION: EMITRANGE Position

Definition at line 902 of file core_cm4.h.

#### 6.71.2.55 #define DWT_FUNCTION_FUNCTION_Msk (0xFUL << DWT_FUNCTION_FUNCTION_Pos)

DWT FUNCTION: FUNCTION Mask

Definition at line 906 of file core_cm4.h.

#### 6.71.2.56 #define DWT_FUNCTION_FUNCTION_Pos 0

DWT FUNCTION: FUNCTION Position

Definition at line 905 of file core_cm4.h.

#### 6.71.2.57 #define DWT_FUNCTION_LNK1ENA_Msk (0x1UL << DWT_FUNCTION_LNK1ENA_Pos)

DWT FUNCTION: LNK1ENA Mask

Definition at line 894 of file core_cm4.h.

#### 6.71.2.58 #define DWT_FUNCTION_LNK1ENA_Pos 9

DWT FUNCTION: LNK1ENA Position

Definition at line 893 of file core_cm4.h.

#### 6.71.2.59 #define DWT_FUNCTION_MATCHED_Msk (0x1UL << DWT_FUNCTION_MATCHED_Pos)

DWT FUNCTION: MATCHED Mask

Definition at line 882 of file core_cm4.h.

#### 6.71.2.60 #define DWT_FUNCTION_MATCHED_Pos 24

DWT FUNCTION: MATCHED Position

Definition at line 881 of file core_cm4.h.

#### 6.71.2.61 #define DWT_LSUCNT_LSUCNT_Msk (0xFFUL << DWT_LSUCNT_LSUCNT_Pos)

DWT LSUCNT: LSUCNT Mask

Definition at line 870 of file core_cm4.h.

#### 6.71.2.62 #define DWT_LSUCNT_LSUCNT_Pos 0

DWT LSUCNT: LSUCNT Position

Definition at line 869 of file core_cm4.h.

#### 6.71.2.63 #define DWT_MASK_MASK_Msk (0x1FUL << DWT_MASK_MASK_Pos)

DWT MASK: MASK Mask

Definition at line 878 of file core_cm4.h.

#### 6.71.2.64 #define DWT_MASK_MASK_Pos 0

DWT MASK: MASK Position

Definition at line 877 of file core_cm4.h.

#### 6.71.2.65 #define DWT_SLEEPCNT_SLEEPCNT_Msk (0xFFUL << DWT_SLEEPCNT_SLEEPCNT_Pos)

DWT SLEEPCNT: SLEEPCNT Mask

Definition at line 866 of file core_cm4.h.

#### 6.71.2.66 #define DWT_SLEEPCNT_SLEEPCNT_Pos 0

DWT SLEEPCNT: SLEEPCNT Position

Definition at line 865 of file core_cm4.h.

## 6.72 Defines and Type Definitions

### 6.72.1 Detailed Description

Type definitions and defines for Cortex-M processor based devices.

### Modules

- Core Debug Registers (CoreDebug)

  *Cortex-M0+ Core Debug Registers (DCB registers, SHCSR, and DFSR) are only accessible over DAP and not via processor. Therefore they are not covered by the Cortex-M0 header file.*
- Core Definitions

  *Definitions for base addresses, unions, and structures.*
- Data Watchpoint and Trace (DWT)

  *Type definitions for the Data Watchpoint and Trace (DWT)*
- Instrumentation Trace Macrocell (ITM)

  *Type definitions for the Instrumentation Trace Macrocell (ITM)*
- Nested Vectored Interrupt Controller (NVIC)

  *Type definitions for the NVIC Registers.*
- Status and Control Registers

  *Core Register type definitions.*
- System Control Block (SCB)

  *Type definitions for the System Control Block Registers.*
- System Controls not in SCB (SCnSCB)

  *Type definitions for the System Control and ID Register not in the SCB.*
- System Tick Timer (SysTick)

  *Type definitions for the System Timer Registers.*
- Trace Port Interface (TPI)

  *Type definitions for the Trace Port Interface (TPI)*

# 6.73 Functions and Instructions Reference

## 6.73.1 Detailed Description

### Modules

- CMSIS Core Register Access Functions
- ITM Functions

  *Functions that access the ITM debug interface.*
- NVIC Functions

  *Functions that manage interrupts and exceptions via the NVIC.*
- SysTick Functions

  *Functions that configure the System.*

## 6.74 IAR EWARM support in LPCOpen

IAR Embedded Workbench for ARM

## 6.75 ITM Functions

### 6.75.1 Detailed Description

Functions that access the ITM debug interface.

#### Macros

- #define ITM_RXBUFFER_EMPTY 0x5AA55AA5

#### Functions

- __STATIC_INLINE uint32_t ITM_SendChar (uint32_t ch)

  *ITM Send Character.*
- __STATIC_INLINE int32_t ITM_ReceiveChar (void)

  *ITM Receive Character.*
- __STATIC_INLINE int32_t ITM_CheckChar (void)

  *ITM Check Character.*

#### Variables

- volatile int32_t ITM_RxBuffer

### 6.75.2 Macro Definition Documentation

#### 6.75.2.1 #define ITM_RXBUFFER_EMPTY 0x5AA55AA5

Value identifying ITM_RxBuffer is ready for next character.

Definition at line 1704 of file core_cm4.h.

### 6.75.3 Function Documentation

#### 6.75.3.1 __STATIC_INLINE int32_t ITM_CheckChar ( void )

ITM Check Character.

The function checks whether a character is pending for reading in the variable ITM_RxBuffer.

**Returns**

0 No character available.
1 Character available.

Definition at line 1755 of file core_cm4.h.

#### 6.75.3.2 __STATIC_INLINE int32_t ITM_ReceiveChar ( void )

ITM Receive Character.

The function inputs a character via the external variable ITM_RxBuffer.

**Returns**

Received character.
-1 No character pending.

Definition at line 1736 of file core_cm4.h.

#### 6.75.3.3 __STATIC_INLINE uint32_t ITM_SendChar ( uint32_t *ch* )

ITM Send Character.

The function transmits a character via the ITM channel 0, and

- Just returns when no debugger is connected that has booked the output.
- Is blocking when a debugger is connected, but the previous character sent has not been transmitted.

**Parameters**

| | | |
|---|---|---|
| in | *ch* | Character to transmit. |

**Returns**

Character to transmit.

Definition at line 1717 of file core_cm4.h.

### 6.75.4 Variable Documentation

#### 6.75.4.1 volatile int32_t ITM_RxBuffer

External variable to receive characters.

## 6.76 Instrumentation Trace Macrocell (ITM)

### 6.76.1 Detailed Description

Type definitions for the Instrumentation Trace Macrocell (ITM)

### Data Structures

- struct ITM_Type

  *Structure type to access the Instrumentation Trace Macrocell Register (ITM).*

### Macros

- #define ITM_TPR_PRIVMASK_Pos 0
- #define ITM_TPR_PRIVMASK_Msk (0xFUL << ITM_TPR_PRIVMASK_Pos)
- #define ITM_TCR_BUSY_Pos 23
- #define ITM_TCR_BUSY_Msk (1UL << ITM_TCR_BUSY_Pos)
- #define ITM_TCR_TraceBusID_Pos 16
- #define ITM_TCR_TraceBusID_Msk (0x7FUL << ITM_TCR_TraceBusID_Pos)
- #define ITM_TCR_GTSFREQ_Pos 10
- #define ITM_TCR_GTSFREQ_Msk (3UL << ITM_TCR_GTSFREQ_Pos)
- #define ITM_TCR_TSPrescale_Pos 8
- #define ITM_TCR_TSPrescale_Msk (3UL << ITM_TCR_TSPrescale_Pos)
- #define ITM_TCR_SWOENA_Pos 4
- #define ITM_TCR_SWOENA_Msk (1UL << ITM_TCR_SWOENA_Pos)
- #define ITM_TCR_DWTENA_Pos 3
- #define ITM_TCR_DWTENA_Msk (1UL << ITM_TCR_DWTENA_Pos)
- #define ITM_TCR_SYNCENA_Pos 2
- #define ITM_TCR_SYNCENA_Msk (1UL << ITM_TCR_SYNCENA_Pos)
- #define ITM_TCR_TSENA_Pos 1
- #define ITM_TCR_TSENA_Msk (1UL << ITM_TCR_TSENA_Pos)
- #define ITM_TCR_ITMENA_Pos 0
- #define ITM_TCR_ITMENA_Msk (1UL << ITM_TCR_ITMENA_Pos)
- #define ITM_IWR_ATVALIDM_Pos 0
- #define ITM_IWR_ATVALIDM_Msk (1UL << ITM_IWR_ATVALIDM_Pos)
- #define ITM_IRR_ATREADYM_Pos 0
- #define ITM_IRR_ATREADYM_Msk (1UL << ITM_IRR_ATREADYM_Pos)
- #define ITM_IMCR_INTEGRATION_Pos 0
- #define ITM_IMCR_INTEGRATION_Msk (1UL << ITM_IMCR_INTEGRATION_Pos)
- #define ITM_LSR_ByteAcc_Pos 2
- #define ITM_LSR_ByteAcc_Msk (1UL << ITM_LSR_ByteAcc_Pos)
- #define ITM_LSR_Access_Pos 1
- #define ITM_LSR_Access_Msk (1UL << ITM_LSR_Access_Pos)
- #define ITM_LSR_Present_Pos 0
- #define ITM_LSR_Present_Msk (1UL << ITM_LSR_Present_Pos)

### 6.76.2 Macro Definition Documentation

#### 6.76.2.1 #define ITM_IMCR_INTEGRATION_Msk (1UL << ITM_IMCR_INTEGRATION_Pos)

ITM IMCR: INTEGRATION Mask

Definition at line 751 of file core_cm4.h.

#### 6.76.2.2 #define ITM_IMCR_INTEGRATION_Pos 0

ITM IMCR: INTEGRATION Position

Definition at line 750 of file core_cm4.h.

#### 6.76.2.3 #define ITM_IRR_ATREADYM_Msk (1UL << ITM_IRR_ATREADYM_Pos)

ITM IRR: ATREADYM Mask

Definition at line 747 of file core_cm4.h.

#### 6.76.2.4 #define ITM_IRR_ATREADYM_Pos 0

ITM IRR: ATREADYM Position

Definition at line 746 of file core_cm4.h.

#### 6.76.2.5 #define ITM_IWR_ATVALIDM_Msk (1UL << ITM_IWR_ATVALIDM_Pos)

ITM IWR: ATVALIDM Mask

Definition at line 743 of file core_cm4.h.

#### 6.76.2.6 #define ITM_IWR_ATVALIDM_Pos 0

ITM IWR: ATVALIDM Position

Definition at line 742 of file core_cm4.h.

#### 6.76.2.7 #define ITM_LSR_Access_Msk (1UL << ITM_LSR_Access_Pos)

ITM LSR: Access Mask

Definition at line 758 of file core_cm4.h.

#### 6.76.2.8 #define ITM_LSR_Access_Pos 1

ITM LSR: Access Position

Definition at line 757 of file core_cm4.h.

#### 6.76.2.9 #define ITM_LSR_ByteAcc_Msk (1UL << ITM_LSR_ByteAcc_Pos)

ITM LSR: ByteAcc Mask

Definition at line 755 of file core_cm4.h.

#### 6.76.2.10 #define ITM_LSR_ByteAcc_Pos 2

ITM LSR: ByteAcc Position

Definition at line 754 of file core_cm4.h.

#### 6.76.2.11 #define ITM_LSR_Present_Msk (1UL << ITM_LSR_Present_Pos)

ITM LSR: Present Mask

Definition at line 761 of file core_cm4.h.

#### 6.76.2.12 #define ITM_LSR_Present_Pos 0

ITM LSR: Present Position

Definition at line 760 of file core_cm4.h.

#### 6.76.2.13 #define ITM_TCR_BUSY_Msk (1UL << ITM_TCR_BUSY_Pos)

ITM TCR: BUSY Mask

Definition at line 715 of file core_cm4.h.

#### 6.76.2.14 #define ITM_TCR_BUSY_Pos 23

ITM TCR: BUSY Position

Definition at line 714 of file core_cm4.h.

#### 6.76.2.15 #define ITM_TCR_DWTENA_Msk (1UL << ITM_TCR_DWTENA_Pos)

ITM TCR: DWTENA Mask

Definition at line 730 of file core_cm4.h.

#### 6.76.2.16 #define ITM_TCR_DWTENA_Pos 3

ITM TCR: DWTENA Position

Definition at line 729 of file core_cm4.h.

#### 6.76.2.17 #define ITM_TCR_GTSFREQ_Msk (3UL << ITM_TCR_GTSFREQ_Pos)

ITM TCR: Global timestamp frequency Mask

Definition at line 721 of file core_cm4.h.

#### 6.76.2.18 #define ITM_TCR_GTSFREQ_Pos 10

ITM TCR: Global timestamp frequency Position

Definition at line 720 of file core_cm4.h.

#### 6.76.2.19 #define ITM_TCR_ITMENA_Msk (1UL << ITM_TCR_ITMENA_Pos)

ITM TCR: ITM Enable bit Mask

Definition at line 739 of file core_cm4.h.

#### 6.76.2.20 #define ITM_TCR_ITMENA_Pos 0

ITM TCR: ITM Enable bit Position

Definition at line 738 of file core_cm4.h.

#### 6.76.2.21 #define ITM_TCR_SWOENA_Msk (1UL << ITM_TCR_SWOENA_Pos)

ITM TCR: SWOENA Mask

Definition at line 727 of file core_cm4.h.

#### 6.76.2.22 #define ITM_TCR_SWOENA_Pos 4

ITM TCR: SWOENA Position

Definition at line 726 of file core_cm4.h.

#### 6.76.2.23 #define ITM_TCR_SYNCENA_Msk (1UL << ITM_TCR_SYNCENA_Pos)

ITM TCR: SYNCENA Mask

Definition at line 733 of file core_cm4.h.

#### 6.76.2.24 #define ITM_TCR_SYNCENA_Pos 2

ITM TCR: SYNCENA Position

Definition at line 732 of file core_cm4.h.

#### 6.76.2.25 #define ITM_TCR_TraceBusID_Msk (0x7FUL << ITM_TCR_TraceBusID_Pos)

ITM TCR: ATBID Mask

Definition at line 718 of file core_cm4.h.

#### 6.76.2.26 #define ITM_TCR_TraceBusID_Pos 16

ITM TCR: ATBID Position

Definition at line 717 of file core_cm4.h.

#### 6.76.2.27 #define ITM_TCR_TSENA_Msk (1UL << ITM_TCR_TSENA_Pos)

ITM TCR: TSENA Mask

Definition at line 736 of file core_cm4.h.

#### 6.76.2.28 #define ITM_TCR_TSENA_Pos 1

ITM TCR: TSENA Position

Definition at line 735 of file core_cm4.h.

#### 6.76.2.29 #define ITM_TCR_TSPrescale_Msk (3UL << ITM_TCR_TSPrescale_Pos)

ITM TCR: TSPrescale Mask

Definition at line 724 of file core_cm4.h.

#### 6.76.2.30 #define ITM_TCR_TSPrescale_Pos 8

ITM TCR: TSPrescale Position

Definition at line 723 of file core_cm4.h.

#### 6.76.2.31 #define ITM_TPR_PRIVMASK_Msk (0xFUL << ITM_TPR_PRIVMASK_Pos)

ITM TPR: PRIVMASK Mask

Definition at line 711 of file core_cm4.h.

#### 6.76.2.32 #define ITM_TPR_PRIVMASK_Pos 0

ITM TPR: PRIVMASK Position

Definition at line 710 of file core_cm4.h.

## 6.77 Keil uVision support in LPCOpen

`MDK-ARM Microcontroller Development Kit (uVision4)`

# 6.78 LPC Public Macros

## 6.78.1 Detailed Description

### Macros

- #define _BIT(n) (1 << (n))
- #define _SBF(f, v) ((v) << (f))
- #define _BITMASK(field_width) ( _BIT(field_width) - 1)
- #define NULL ((void ∗) 0)
- #define NELEMENTS(array) (sizeof(array) / sizeof(array[0]))
- #define STATIC static
- #define EXTERN extern
- #define MAX(a, b) (((a) > (b)) ? (a) : (b))
- #define MIN(a, b) (((a) < (b)) ? (a) : (b))

## 6.78.2 Macro Definition Documentation

### 6.78.2.1 #define _BIT( *n* ) (1 << (n))

Definition at line 104 of file lpc_types.h.

### 6.78.2.2 #define _BITMASK( *field_width* ) ( _BIT(field_width) - 1)

Definition at line 129 of file lpc_types.h.

### 6.78.2.3 #define _SBF( *f,* *v* ) ((v) << (f))

Definition at line 112 of file lpc_types.h.

### 6.78.2.4 #define EXTERN extern

Definition at line 142 of file lpc_types.h.

### 6.78.2.5 #define MAX( *a,* *b* ) (((a) > (b)) ? (a) : (b))

Definition at line 145 of file lpc_types.h.

### 6.78.2.6 #define MIN( *a,* *b* ) (((a) < (b)) ? (a) : (b))

Definition at line 148 of file lpc_types.h.

### 6.78.2.7 #define NELEMENTS( *array* ) (sizeof(array) / sizeof(array[0]))

Definition at line 137 of file lpc_types.h.

### 6.78.2.8 #define NULL ((void ∗) 0)

Definition at line 133 of file lpc_types.h.

#### 6.78.2.9 #define STATIC static

Definition at line 140 of file lpc_types.h.

# 6.79 LPC Public Types

## 6.79.1 Detailed Description

### Macros

- #define PARAM_SETSTATE(State) ((State == RESET) || (State == SET))
- #define PARAM_FUNCTIONALSTATE(State) ((State == DISABLE) || (State == ENABLE))
- #define INLINE inline
- #define ALIGN(x) __attribute__ ((aligned(x)))
- #define WEAK __attribute__((weak))

### Typedefs

- typedef enum FlagStatus IntStatus
- typedef enum FlagStatus SetState
- typedef void(∗ PFV) ()
- typedef int32_t(∗ PFI) ()
- typedef char CHAR
- typedef uint8_t UNS_8
- typedef int8_t INT_8
- typedef uint16_t UNS_16
- typedef int16_t INT_16
- typedef uint32_t UNS_32
- typedef int32_t INT_32
- typedef int64_t INT_64
- typedef uint64_t UNS_64
- typedef bool BOOL_32
- typedef bool BOOL_16
- typedef bool BOOL_8

### Enumerations

- enum Bool { FALSE = 0, TRUE = !FALSE }

  *Boolean Type definition.*
- enum FlagStatus { RESET = 0, SET = !RESET }

  *Boolean Type definition.*
- enum FunctionalState { DISABLE = 0, ENABLE = !DISABLE }

  *Functional State Definition.*
- enum Status { ERROR = 0, SUCCESS = !ERROR }
- enum TRANSFER_BLOCK_T { NONE_BLOCKING = 0, BLOCKING }

## 6.79.2 Macro Definition Documentation

### 6.79.2.1 #define ALIGN( *x* ) __attribute__ ((aligned(x)))

Definition at line 213 of file lpc_types.h.

### 6.79.2.2 #define INLINE inline

Definition at line 205 of file lpc_types.h.

#### 6.79.2.3 #define PARAM_FUNCTIONALSTATE( *State* ) ((State == DISABLE) || (State == ENABLE))

Definition at line 69 of file lpc_types.h.

#### 6.79.2.4 #define PARAM_SETSTATE( *State* ) ((State == RESET) || (State == SET))

Definition at line 63 of file lpc_types.h.

#### 6.79.2.5 #define WEAK __attribute__((weak))

Definition at line 214 of file lpc_types.h.

### 6.79.3 Typedef Documentation

#### 6.79.3.1 typedef bool BOOL_16

16 bit boolean type

Definition at line 196 of file lpc_types.h.

#### 6.79.3.2 typedef bool BOOL_32

32 bit boolean type

Definition at line 193 of file lpc_types.h.

#### 6.79.3.3 typedef bool BOOL_8

8 bit boolean type

Definition at line 199 of file lpc_types.h.

#### 6.79.3.4 typedef char CHAR

LPC type for character type

Definition at line 161 of file lpc_types.h.

#### 6.79.3.5 typedef int16_t INT_16

LPC type for 16 bit signed value

Definition at line 173 of file lpc_types.h.

#### 6.79.3.6 typedef int32_t INT_32

LPC type for 32 bit signed value

Definition at line 179 of file lpc_types.h.

#### 6.79.3.7 typedef int64_t INT_64

LPC type for 64 bit signed value

Definition at line 182 of file lpc_types.h.

#### 6.79.3.8 typedef int8_t INT_8

LPC type for 8 bit signed value

Definition at line 167 of file lpc_types.h.

#### 6.79.3.9 typedef enum FlagStatus IntStatus

#### 6.79.3.10 typedef int32_t(∗ PFI) ()

Pointer to Function returning int32_t (any number of parameters)

Definition at line 88 of file lpc_types.h.

#### 6.79.3.11 typedef void(∗ PFV) ()

Pointer to Function returning Void (any number of parameters)

Definition at line 85 of file lpc_types.h.

#### 6.79.3.12 typedef enum FlagStatus SetState

#### 6.79.3.13 typedef uint16_t UNS_16

LPC type for 16 bit unsigned value

Definition at line 170 of file lpc_types.h.

#### 6.79.3.14 typedef uint32_t UNS_32

LPC type for 32 bit unsigned value

Definition at line 176 of file lpc_types.h.

#### 6.79.3.15 typedef uint64_t UNS_64

LPC type for 64 bit unsigned value

Definition at line 185 of file lpc_types.h.

#### 6.79.3.16 typedef uint8_t UNS_8

LPC type for 8 bit unsigned value

Definition at line 164 of file lpc_types.h.

### 6.79.4 Enumeration Type Documentation

#### 6.79.4.1 enum Bool

Boolean Type definition.

**Enumerator**

***FALSE***

***TRUE***

Definition at line 50 of file lpc_types.h.

#### 6.79.4.2 enum FlagStatus

Boolean Type definition.

Flag Status and Interrupt Flag Status type definition

**Enumerator**

- ***RESET***
- ***SET***

Definition at line 62 of file lpc_types.h.

#### 6.79.4.3 enum FunctionalState

Functional State Definition.

**Enumerator**

- ***DISABLE***
- ***ENABLE***

Definition at line 68 of file lpc_types.h.

#### 6.79.4.4 enum Status

@ Status type definition

**Enumerator**

- ***ERROR***
- ***SUCCESS***

Definition at line 74 of file lpc_types.h.

#### 6.79.4.5 enum TRANSFER_BLOCK_T

Read/Write transfer type mode (Block or non-block)

**Enumerator**

| | |
|---|---|
| ***NONE_BLOCKING*** | None Blocking type |
| ***BLOCKING*** | Blocking type |

Definition at line 79 of file lpc_types.h.

## 6.80 LPC5411X multi-core use in LPCOpen

The LPC5411x device family has both a Cortex M4F and a Cortex M0+ CPU core that can execute applications simultaneously and signal events and status to the other core via the mailbox peripheral. One core is configured as the system master - usually the M4F core, while the other core is configured as the system slave. On system reset, the master core boots to FLASH at address 0x00000000, while the slave core sleeps until it is restarted by the master core. When the slave core is next reset by the master core application, it boots into the startup code at address 0x00000000. *Because both the M4F and M0 cores boot into the same FLASH address, the startup code must work correctly on both cores. The LPCOpen startup code that boots in FLASH is written using CM0+ instructions only and works on both CM4F and CM0+ cores.*

**LPC5411x master and slave core boot processes**
Either core can be the master or slave core depending on the configuration of your device (part number). In some parts, only one core may be available (single core).

On system reset, both cores boot into the code at address 0x00000000 at the same time. For this initial boot after reset, ROM is mapped to address 0x000000000. The ROM code will eventually remap FLASH to address 0x00000000, boot the master core in FLASH address 0x00000000, and idle the slave core in ROM using the 'Wait For Interrupt' (WFI) instruction. The slave will remain in the WFI state until reset by the master core.

**LPCOpen startup code**
LPCOpen startup code placed at FLASH address 0x00000000 is used for both the master and slave core. The startup code, when run on the master core, will start the execution of the application immediately - usually SystemInit() and then main(). The startup code, when run on the slave core, will look at the special execution and stack address provided by the master core. If the execution address is non-0x0, the stack address will be loaded from the value provided by the master core and then execution will be handed off to the address provided by he master core. If the execution address is 0x0, the slave code will sleep via WFI in the startup code in FLASH.

The core type - either M4F or M0+ - and the master or slave configuration can be determined at run-time for an application using the following functions:

```
/**
 * @brief Determinc which MCU this code is running on
 * @return  true if executing on the CM4, or false if executing on the CM0+
 */
bool Chip_CPU_IsM4Core(void);

/**
 * @brief Determine if this core is a slave or master
 * @return  true if this MCU is operating as the master, or false if operating as a slave
 */
bool Chip_CPU_IsMasterCore(void);
@endverbtaim
 *
 * The slave core can be reset and booted with a custom execution and stack
 * address with the following functions:<br>
@verbatim
/**
 * @brief Setup M0+ boot and reset M0+ core
 * @param coentry  : Pointer to boot entry point for M0+ core
 * @param costackptr : Pointer to where stack should be located for M0+ core
 * @return  Nothing
 * @note Will setup boot stack and entry point, enable M0+ clock and then
 * reset M0+ core.
 */
void Chip_CPU_CM0Boot(uint32_t *coentry, uint32_t *costackptr);

/**
 * @brief Setup M4 boot and reset M4 core
 * @param coentry  : Pointer to boot entry point for M4 core
 * @param costackptr : Pointer to where stack should be located for M4 core
 * @return  Nothing
 * @note Will setup boot stack and entry point, enable M4 clock and then
 * reset M0+ core.
 */
void Chip_CPU_CM4Boot(uint32_t *coentry, uint32_t *costackptr);
```

***Special notes:***
***Do not use the core boot functions to boot the same core the function is running on - it will not work! All of the multi-core support functions are part of the powerlib library.***

The example code below safely boots the M0+ core from the M4F core, but can be executed on both cores safely.

```
/* Do not boot if on slave core */
if (Chip_CPU_IsMasterCore() == true) {
 /* Boot 'other' core, whichever is the slave */
 if (Chip_CPU_IsM4Core()) {
  /* This is M4 and master, so boot M0+ */
  Chip_CPU_CM0Boot(bootAddr, stackAddr);
 }
}
```

**Debugger settings for IAR Embedded Workbench and Keil ARM MDK**
The images below show the necessary settings to debug the images in the M4F and M0+ cores. *LPCXpresso is not shown, but provides integrated support in the tool for both cores.*

*Debugger settings for both cores are almost exactly the same except for the CPU numner for the target. Keil refers to this selection as "AP" in the CMSIS-DAP debugger dialog, while IAR uses "CPU number on target" on it's CMSIS-DAP selection dialog. The name and description of this option may vary between tool chains and debuggers. The M4F core should use a value of '0', while the M0+ core should use a value of '1'.*

**IAR Embedded Workbench Cortex M4F debugger settings**
The image below shows the debugger setup specific to CMSIS-DAP and the M4F core. Other debuggers setup in IAR are similar.

**IAR Embedded Workbench Cortex M0+ debugger settings**
The image below shows the debugger setup specific to CMSIS-DAP and the M0+ core. Other debuggers setup in IAR are similar.

**Keil ARM MDK Cortex M4F debugger settings**
The image below shows the debugger setup specific to CMSIS-DAP and the M4F core. Other debuggers setup in Keil are similar.

**Keil ARM MDK Cortex M0+ debugger settings**
The image below shows the debugger setup specific to CMSIS-DAP and the M0+ core. Other debuggers setup in Keil are similar.

## 6.81 LPC5411x Chip specific drivers

### 6.81.1 Detailed Description

### Modules

- CHIP: LPC5411X A/D conversion driver
- CHIP: LPC5410x family CMSIS include files
- CHIP: LPC5411X 32-bit Timer driver
- CHIP: LPC5411X CPU multi-core support driver
- CHIP: LPC5411X Clock Driver
- CHIP: LPC5411X Cyclic Redundancy Check Engine driver
- CHIP: LPC5411X DMA Engine driver (legacy)
- CHIP: LPC5411X DMA Service driver
- CHIP: LPC5411X DMIC driver
- CHIP: LPC5411X Enhanced boot block support
- CHIP: LPC5411X GPIO driver
- CHIP: LPC5411X GPIO group driver
- CHIP: LPC5411X IOCON register block and driver
- CHIP: LPC5411X Input Mux Registers and Driver
- CHIP: LPC5411X Mailbox M4/M0+ driver
- CHIP: LPC5411X Micro Tick driver
- CHIP: LPC5411X Multi-Rate Timer driver
- CHIP: LPC5411X PLL Driver
- CHIP: LPC5411X Peripheral addresses and register set declarations
- CHIP: LPC5411X Pin Interrupt and Pattern Match driver
- CHIP: LPC5411X Power LIBRARY functions
- CHIP: LPC5411X Power Management declarations and functions
- CHIP: LPC5411X ROM API declarations and functions
- CHIP: LPC5411X Real Time clock
- CHIP: LPC5411X SPI driver
- CHIP: LPC5411X State Configurable Timer PWM driver
- CHIP: LPC5411X State Configurable Timer driver
- CHIP: LPC5411X System and Control Driver
- CHIP: LPC5411X UART Driver
- CHIP: LPC5411X Windowed Watchdog driver
- CHIP: LPC5411X flexcomm API
- CHIP: LPC5411X support functions
- CHIP: LPC5411x Chip driver build time options
- CHIP: LPC5411x I2C driver
- LPC5411X multi-core use in LPCOpen

## 6.82 LPCOpen download and installation information

### 6.82.1 Detailed Description

**LPCOpen source and projects package**
The LPCOpen platform source and project packages are released as a ZIP files targeted for specific platforms and toolchains. The latest version of the LPCopen package can be download at:
`LPCOpen software package page on LPCware.com`

**LPCOpen software API documentation**
The LPCOpen software API documentation package consists of descriptions of examples and software driver APIs. The LPCOpen LPCOpen software API documentation packages are available online as a separate downloadable ZIP file that is not part of the LPCOpen source and projects package.
`LPCOpen documentation package page on LPCware.com`

**LPCOpen installation - PLEASE READ THIS!**
FIXME - need install insts for flat releases and LPCXpresso releases - covered by QS guides? Installing LPCOpen is simple! Just unzip the LPCOpen ZIP file somewhere on your host system!
All projects and examples are path relative and will automatically adapt to where the files are installed.
***It is VERY HIGHLY recommended to install LPCOpen close to the root directory, as the arguments passed to the build tools can be get fairly long and cause environment issues. If you install the LPCopen platform directory too deep in your filesystem, the paths passed as arguments to build programs may exceed the limit for passed arguments.***
***The recommended installation directory is: c:/LPCOpen***

**Optional components needed for some LPCOpen examples**
emWin download and installation

**LPCOpen release history**
LPCOpen versioning and release history

### Modules

- emWin download and installation

## 6.83 LPCOpen versioning and release history

**For history and known issues, please check out the comprehensive version and history page at** `www.LPCware.com`

# 6.84 NVIC Functions

## 6.84.1 Detailed Description

Functions that manage interrupts and exceptions via the NVIC.

### Macros

- #define _BIT_SHIFT(IRQn) ( (((uint32_t)(IRQn) ) & 0x03) ∗ 8 )
- #define _SHP_IDX(IRQn) ( ((((uint32_t)(IRQn) & 0x0F)-8) >> 2) )
- #define _IP_IDX(IRQn) ( ((uint32_t)(IRQn) >> 2) )

### Functions

- __STATIC_INLINE void NVIC_EnableIRQ (IRQn_Type IRQn)

  *Enable External Interrupt.*
- __STATIC_INLINE void NVIC_DisableIRQ (IRQn_Type IRQn)

  *Disable External Interrupt.*
- __STATIC_INLINE uint32_t NVIC_GetPendingIRQ (IRQn_Type IRQn)

  *Get Pending Interrupt.*
- __STATIC_INLINE void NVIC_SetPendingIRQ (IRQn_Type IRQn)

  *Set Pending Interrupt.*
- __STATIC_INLINE void NVIC_ClearPendingIRQ (IRQn_Type IRQn)

  *Clear Pending Interrupt.*
- __STATIC_INLINE void NVIC_SetPriority (IRQn_Type IRQn, uint32_t priority)

  *Set Interrupt Priority.*
- __STATIC_INLINE uint32_t NVIC_GetPriority (IRQn_Type IRQn)

  *Get Interrupt Priority.*
- __STATIC_INLINE void NVIC_SystemReset (void)

  *System Reset.*
- __STATIC_INLINE void NVIC_SetPriorityGrouping (uint32_t PriorityGroup)

  *Set Priority Grouping.*
- __STATIC_INLINE uint32_t NVIC_GetPriorityGrouping (void)

  *Get Priority Grouping.*
- __STATIC_INLINE uint32_t NVIC_GetActive (IRQn_Type IRQn)

  *Get Active Interrupt.*
- __STATIC_INLINE uint32_t NVIC_EncodePriority (uint32_t PriorityGroup, uint32_t PreemptPriority, uint32_t SubPriority)

  *Encode Priority.*
- __STATIC_INLINE void NVIC_DecodePriority (uint32_t Priority, uint32_t PriorityGroup, uint32_t ∗pPreemptPriority, uint32_t ∗pSubPriority)

  *Decode Priority.*

## 6.84.2 Macro Definition Documentation

### 6.84.2.1 #define _BIT_SHIFT( *IRQn* ) ( (((uint32_t)(IRQn) ) & 0x03) ∗ 8 )

Definition at line 615 of file core_cm0plus.h.

#### 6.84.2.2 #define _IP_IDX( *IRQn* ) ( ((uint32_t)(IRQn) >> 2) )

Definition at line 617 of file core_cm0plus.h.

#### 6.84.2.3 #define _SHP_IDX( *IRQn* ) ( ((((uint32_t)(IRQn) & 0x0F)-8) >> 2) )

Definition at line 616 of file core_cm0plus.h.

### 6.84.3 Function Documentation

#### 6.84.3.1 __STATIC_INLINE void NVIC_ClearPendingIRQ ( IRQn_Type *IRQn* )

Clear Pending Interrupt.

The function clears the pending bit of an external interrupt.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | External interrupt number. Value cannot be negative. |

Definition at line 678 of file core_cm0plus.h.

#### 6.84.3.2 __STATIC_INLINE void NVIC_DecodePriority ( uint32_t *Priority,* uint32_t *PriorityGroup,* uint32_t * *pPreemptPriority,* uint32_t * *pSubPriority* )

Decode Priority.

The function decodes an interrupt priority value with a given priority group to preemptive priority value and subpriority value. In case of a conflict between priority grouping and available priority bits (__NVIC_PRIO_BITS) the samllest possible priority group is set.

**Parameters**

| | | |
|---|---|---|
| in | *Priority* | Priority value, which can be retrieved with the function NVIC_GetPriority(). |
| in | *PriorityGroup* | Used priority group. |
| out | *pPreemptPriority* | Preemptive priority value (starting from 0). |
| out | *pSubPriority* | Subpriority value (starting from 0). |

Definition at line 1620 of file core_cm4.h.

#### 6.84.3.3 __STATIC_INLINE void NVIC_DisableIRQ ( IRQn_Type *IRQn* )

Disable External Interrupt.

The function disables a device-specific interrupt in the NVIC interrupt controller.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | External interrupt number. Value cannot be negative. |

Definition at line 638 of file core_cm0plus.h.

#### 6.84.3.4 __STATIC_INLINE void NVIC_EnableIRQ ( IRQn_Type *IRQn* )

Enable External Interrupt.

The function enables a device-specific interrupt in the NVIC interrupt controller.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | External interrupt number. Value cannot be negative. |

Definition at line 626 of file core_cm0plus.h.

#### 6.84.3.5 __STATIC_INLINE uint32_t NVIC_EncodePriority ( uint32_t *PriorityGroup,* uint32_t *PreemptPriority,* uint32_t *SubPriority* )

Encode Priority.

The function encodes the priority for an interrupt with the given priority group, preemptive priority value, and subpriority value. In case of a conflict between priority grouping and available priority bits (__NVIC_PRIO_BITS), the samllest possible priority group is set.

**Parameters**

| | | |
|---|---|---|
| in | *PriorityGroup* | Used priority group. |
| in | *PreemptPriority* | Preemptive priority value (starting from 0). |
| in | *SubPriority* | Subpriority value (starting from 0). |

**Returns**

Encoded priority. Value can be used in the function NVIC_SetPriority().

Definition at line 1592 of file core_cm4.h.

#### 6.84.3.6 __STATIC_INLINE uint32_t NVIC_GetActive ( IRQn_Type *IRQn* )

Get Active Interrupt.

The function reads the active register in NVIC and returns the active bit.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | Interrupt number. |

**Returns**

0 Interrupt status is not active.
1 Interrupt status is active.

Definition at line 1535 of file core_cm4.h.

#### 6.84.3.7 __STATIC_INLINE uint32_t NVIC_GetPendingIRQ ( IRQn_Type *IRQn* )

Get Pending Interrupt.

The function reads the pending register in the NVIC and returns the pending bit for the specified interrupt.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | Interrupt number. |

**Returns**

0 Interrupt status is not pending.
1 Interrupt status is pending.

Definition at line 654 of file core_cm0plus.h.

#### 6.84.3.8 __STATIC_INLINE uint32_t NVIC_GetPriority ( IRQn_Type *IRQn* )

Get Interrupt Priority.

The function reads the priority of an interrupt. The interrupt number can be positive to specify an external (device specific) interrupt, or negative to specify an internal (core) interrupt.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | Interrupt number. |

**Returns**

Interrupt Priority. Value is aligned automatically to the implemented priority bits of the microcontroller.

Definition at line 715 of file core_cm0plus.h.

#### 6.84.3.9 __STATIC_INLINE uint32_t NVIC_GetPriorityGrouping ( void )

Get Priority Grouping.

The function reads the priority grouping field from the NVIC Interrupt Controller.

**Returns**

Priority grouping field (SCB->AIRCR [10:8] PRIGROUP field).

Definition at line 1455 of file core_cm4.h.

#### 6.84.3.10 __STATIC_INLINE void NVIC_SetPendingIRQ ( IRQn_Type *IRQn* )

Set Pending Interrupt.

The function sets the pending bit of an external interrupt.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | Interrupt number. Value cannot be negative. |

Definition at line 666 of file core_cm0plus.h.

#### 6.84.3.11 __STATIC_INLINE void NVIC_SetPriority ( IRQn_Type *IRQn,* uint32_t *priority* )

Set Interrupt Priority.

The function sets the priority of an interrupt.

**Note**

The priority cannot be set for every core interrupt.

**Parameters**

| | | |
|---|---|---|
| in | *IRQn* | Interrupt number. |
| in | *priority* | Priority to set. |

Definition at line 693 of file core_cm0plus.h.

#### 6.84.3.12 __STATIC_INLINE void NVIC_SetPriorityGrouping ( uint32_t *PriorityGroup* )

Set Priority Grouping.

The function sets the priority grouping field using the required unlock sequence. The parameter PriorityGroup is assigned to the field SCB->AIRCR [10:8] PRIGROUP field. Only values from 0..7 are used. In case of a conflict between priority grouping and available priority bits (__NVIC_PRIO_BITS), the smallest possible priority group is set.

**Parameters**

| | | |
|---|---|---|
| in | *PriorityGroup* | Priority grouping field. |

Definition at line 1435 of file core_cm4.h.

#### 6.84.3.13 __STATIC_INLINE void NVIC_SystemReset ( void )

System Reset.

The function initiates a system reset request to reset the MCU.

Definition at line 729 of file core_cm0plus.h.

## 6.85 NXP LPCXpresso LPC54114 LQFP board

**Documentation links**
LPCXpresso Boards Overview

**Board software Support**
BOARD_LPCXPRESSO_54114

**Jumper configuration**
The recommended jumper configuration for the LPCXpresso LQFP LPC54114 is below:
JP1 open
JP2 shorted on LOC pins (pins 1-2)
P1 open
JP5 open (see CMSIS-DAP/BRIDGE firmware installation)
JP6 open
JP9 on pins 2-3
JP10 shorted

**Board recovery using the ISP pin**
If the board's FLASH can no longer be programmed, the image in FLASH may be corrupt. To recover the board, the board must be reset with the ISP pin held low. To do this, reset and/or program the board with switch SW2 (ISP0) held down. The switch can be released once programming starts.

# 6.86 Nested Vectored Interrupt Controller (NVIC)

## 6.86.1 Detailed Description

Type definitions for the NVIC Registers.

### Data Structures

- struct NVIC_Type

  *Structure type to access the Nested Vectored Interrupt Controller (NVIC).*

### Macros

- #define NVIC_STIR_INTID_Pos 0
- #define NVIC_STIR_INTID_Msk (0x1FFUL << NVIC_STIR_INTID_Pos)

## 6.86.2 Macro Definition Documentation

### 6.86.2.1 #define NVIC_STIR_INTID_Msk (0x1FFUL << NVIC_STIR_INTID_Pos)

STIR: INTLINESNUM Mask

Definition at line 355 of file core_cm4.h.

### 6.86.2.2 #define NVIC_STIR_INTID_Pos 0

STIR: INTLINESNUM Position

Definition at line 354 of file core_cm4.h.

## 6.87 RTOS support code

### 6.87.1 Detailed Description

RTOS support code includes custom drivers or support files for various Real-time operating systems and NXP devices.

### Modules

- RTOS: FreeRTOS support code

## 6.88 RTOS: FreeRTOS support code

### 6.88.1 Detailed Description

`FreeRTOS` an open-source RTOS

### Modules

- Common FreeRTOS functions shared with multiple platforms

## 6.89 Status and Control Registers

### 6.89.1 Detailed Description

Core Register type definitions.

### Data Structures

- union APSR_Type

  *Union type to access the Application Program Status Register (APSR).*
- union IPSR_Type

  *Union type to access the Interrupt Program Status Register (IPSR).*
- union xPSR_Type

  *Union type to access the Special-Purpose Program Status Registers (xPSR).*
- union CONTROL_Type

  *Union type to access the Control Registers (CONTROL).*

# 6.90 Supported toolchains in LPCOpen

## 6.90.1 Detailed Description

The LPCOpen platform supports the following toolchains. Not all examples support all toolchains. See the example pages for more information about which toolchains are supported for specific examples.

Quickstart guides can be found online at the LPCOpen quickstart `page.`

Keil uVision support in LPCOpen IAR EWARM support in LPCOpen Code Red LPCXpresso support in LPCOpen

### Modules

- Code Red LPCXpresso support in LPCOpen
- IAR EWARM support in LPCOpen
- Keil uVision support in LPCOpen

## 6.91 SysTick Functions

### 6.91.1 Detailed Description

Functions that configure the System.

### Functions

- __STATIC_INLINE uint32_t SysTick_Config (uint32_t ticks)

  *System Tick Configuration.*

### 6.91.2 Function Documentation

#### 6.91.2.1 __STATIC_INLINE uint32_t SysTick_Config ( uint32_t *ticks* )

System Tick Configuration.

The function initializes the System Timer and its interrupt, and starts the System Tick Timer. Counter is in free running mode to generate periodic interrupts.

**Parameters**

| | | |
|---|---|---|
| `in` | *ticks* | Number of ticks between two interrupts. |

**Returns**

0 Function succeeded.
1 Function failed.

**Note**

When the variable **__Vendor_SysTickConfig** is set to 1, then the function **SysTick_Config** is not included. In this case, the file ***device*.h** must contain a vendor-specific implementation of this function.

Definition at line 767 of file core_cm0plus.h.

## 6.92 System Control Block (SCB)

### 6.92.1 Detailed Description

Type definitions for the System Control Block Registers.

### Data Structures

- struct SCB_Type

  *Structure type to access the System Control Block (SCB).*

### Macros

- #define SCB_CPUID_IMPLEMENTER_Pos 24
- #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
- #define SCB_CPUID_VARIANT_Pos 20
- #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)
- #define SCB_CPUID_ARCHITECTURE_Pos 16
- #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
- #define SCB_CPUID_PARTNO_Pos 4
- #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)
- #define SCB_CPUID_REVISION_Pos 0
- #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)
- #define SCB_ICSR_NMIPENDSET_Pos 31
- #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)
- #define SCB_ICSR_PENDSVSET_Pos 28
- #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)
- #define SCB_ICSR_PENDSVCLR_Pos 27
- #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)
- #define SCB_ICSR_PENDSTSET_Pos 26
- #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)
- #define SCB_ICSR_PENDSTCLR_Pos 25
- #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)
- #define SCB_ICSR_ISRPREEMPT_Pos 23
- #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)
- #define SCB_ICSR_ISRPENDING_Pos 22
- #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)
- #define SCB_ICSR_VECTPENDING_Pos 12
- #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)
- #define SCB_ICSR_VECTACTIVE_Pos 0
- #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)
- #define SCB_AIRCR_VECTKEY_Pos 16
- #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)
- #define SCB_AIRCR_VECTKEYSTAT_Pos 16
- #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)
- #define SCB_AIRCR_ENDIANESS_Pos 15
- #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)
- #define SCB_AIRCR_SYSRESETREQ_Pos 2
- #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)
- #define SCB_AIRCR_VECTCLRACTIVE_Pos 1
- #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)
- #define SCB_SCR_SEVONPEND_Pos 4
- #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)

- #define SCB_SCR_SLEEPDEEP_Pos 2
- #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)
- #define SCB_SCR_SLEEPONEXIT_Pos 1
- #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)
- #define SCB_CCR_STKALIGN_Pos 9
- #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)
- #define SCB_CCR_UNALIGN_TRP_Pos 3
- #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)
- #define SCB_SHCSR_SVCALLPENDED_Pos 15
- #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)
- #define SCB_CPUID_IMPLEMENTER_Pos 24
- #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
- #define SCB_CPUID_VARIANT_Pos 20
- #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)
- #define SCB_CPUID_ARCHITECTURE_Pos 16
- #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
- #define SCB_CPUID_PARTNO_Pos 4
- #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)
- #define SCB_CPUID_REVISION_Pos 0
- #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)
- #define SCB_ICSR_NMIPENDSET_Pos 31
- #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)
- #define SCB_ICSR_PENDSVSET_Pos 28
- #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)
- #define SCB_ICSR_PENDSVCLR_Pos 27
- #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)
- #define SCB_ICSR_PENDSTSET_Pos 26
- #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)
- #define SCB_ICSR_PENDSTCLR_Pos 25
- #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)
- #define SCB_ICSR_ISRPREEMPT_Pos 23
- #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)
- #define SCB_ICSR_ISRPENDING_Pos 22
- #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)
- #define SCB_ICSR_VECTPENDING_Pos 12
- #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)
- #define SCB_ICSR_RETTOBASE_Pos 11
- #define SCB_ICSR_RETTOBASE_Msk (1UL << SCB_ICSR_RETTOBASE_Pos)
- #define SCB_ICSR_VECTACTIVE_Pos 0
- #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)
- #define SCB_VTOR_TBLOFF_Pos 7
- #define SCB_VTOR_TBLOFF_Msk (0x1FFFFFFUL << SCB_VTOR_TBLOFF_Pos)
- #define SCB_AIRCR_VECTKEY_Pos 16
- #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)
- #define SCB_AIRCR_VECTKEYSTAT_Pos 16
- #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)
- #define SCB_AIRCR_ENDIANESS_Pos 15
- #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)
- #define SCB_AIRCR_PRIGROUP_Pos 8
- #define SCB_AIRCR_PRIGROUP_Msk (7UL << SCB_AIRCR_PRIGROUP_Pos)
- #define SCB_AIRCR_SYSRESETREQ_Pos 2
- #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)
- #define SCB_AIRCR_VECTCLRACTIVE_Pos 1
- #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)
- #define SCB_AIRCR_VECTRESET_Pos 0

- #define SCB_AIRCR_VECTRESET_Msk (1UL << SCB_AIRCR_VECTRESET_Pos)
- #define SCB_SCR_SEVONPEND_Pos 4
- #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)
- #define SCB_SCR_SLEEPDEEP_Pos 2
- #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)
- #define SCB_SCR_SLEEPONEXIT_Pos 1
- #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)
- #define SCB_CCR_STKALIGN_Pos 9
- #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)
- #define SCB_CCR_BFHFNMIGN_Pos 8
- #define SCB_CCR_BFHFNMIGN_Msk (1UL << SCB_CCR_BFHFNMIGN_Pos)
- #define SCB_CCR_DIV_0_TRP_Pos 4
- #define SCB_CCR_DIV_0_TRP_Msk (1UL << SCB_CCR_DIV_0_TRP_Pos)
- #define SCB_CCR_UNALIGN_TRP_Pos 3
- #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)
- #define SCB_CCR_USERSETMPEND_Pos 1
- #define SCB_CCR_USERSETMPEND_Msk (1UL << SCB_CCR_USERSETMPEND_Pos)
- #define SCB_CCR_NONBASETHRDENA_Pos 0
- #define SCB_CCR_NONBASETHRDENA_Msk (1UL << SCB_CCR_NONBASETHRDENA_Pos)
- #define SCB_SHCSR_USGFAULTENA_Pos 18
- #define SCB_SHCSR_USGFAULTENA_Msk (1UL << SCB_SHCSR_USGFAULTENA_Pos)
- #define SCB_SHCSR_BUSFAULTENA_Pos 17
- #define SCB_SHCSR_BUSFAULTENA_Msk (1UL << SCB_SHCSR_BUSFAULTENA_Pos)
- #define SCB_SHCSR_MEMFAULTENA_Pos 16
- #define SCB_SHCSR_MEMFAULTENA_Msk (1UL << SCB_SHCSR_MEMFAULTENA_Pos)
- #define SCB_SHCSR_SVCALLPENDED_Pos 15
- #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)
- #define SCB_SHCSR_BUSFAULTPENDED_Pos 14
- #define SCB_SHCSR_BUSFAULTPENDED_Msk (1UL << SCB_SHCSR_BUSFAULTPENDED_Pos)
- #define SCB_SHCSR_MEMFAULTPENDED_Pos 13
- #define SCB_SHCSR_MEMFAULTPENDED_Msk (1UL << SCB_SHCSR_MEMFAULTPENDED_Pos)
- #define SCB_SHCSR_USGFAULTPENDED_Pos 12
- #define SCB_SHCSR_USGFAULTPENDED_Msk (1UL << SCB_SHCSR_USGFAULTPENDED_Pos)
- #define SCB_SHCSR_SYSTICKACT_Pos 11
- #define SCB_SHCSR_SYSTICKACT_Msk (1UL << SCB_SHCSR_SYSTICKACT_Pos)
- #define SCB_SHCSR_PENDSVACT_Pos 10
- #define SCB_SHCSR_PENDSVACT_Msk (1UL << SCB_SHCSR_PENDSVACT_Pos)
- #define SCB_SHCSR_MONITORACT_Pos 8
- #define SCB_SHCSR_MONITORACT_Msk (1UL << SCB_SHCSR_MONITORACT_Pos)
- #define SCB_SHCSR_SVCALLACT_Pos 7
- #define SCB_SHCSR_SVCALLACT_Msk (1UL << SCB_SHCSR_SVCALLACT_Pos)
- #define SCB_SHCSR_USGFAULTACT_Pos 3
- #define SCB_SHCSR_USGFAULTACT_Msk (1UL << SCB_SHCSR_USGFAULTACT_Pos)
- #define SCB_SHCSR_BUSFAULTACT_Pos 1
- #define SCB_SHCSR_BUSFAULTACT_Msk (1UL << SCB_SHCSR_BUSFAULTACT_Pos)
- #define SCB_SHCSR_MEMFAULTACT_Pos 0
- #define SCB_SHCSR_MEMFAULTACT_Msk (1UL << SCB_SHCSR_MEMFAULTACT_Pos)
- #define SCB_CFSR_USGFAULTSR_Pos 16
- #define SCB_CFSR_USGFAULTSR_Msk (0xFFFFUL << SCB_CFSR_USGFAULTSR_Pos)
- #define SCB_CFSR_BUSFAULTSR_Pos 8
- #define SCB_CFSR_BUSFAULTSR_Msk (0xFFUL << SCB_CFSR_BUSFAULTSR_Pos)
- #define SCB_CFSR_MEMFAULTSR_Pos 0
- #define SCB_CFSR_MEMFAULTSR_Msk (0xFFUL << SCB_CFSR_MEMFAULTSR_Pos)
- #define SCB_HFSR_DEBUGEVT_Pos 31
- #define SCB_HFSR_DEBUGEVT_Msk (1UL << SCB_HFSR_DEBUGEVT_Pos)

- #define SCB_HFSR_FORCED_Pos 30
- #define SCB_HFSR_FORCED_Msk (1UL << SCB_HFSR_FORCED_Pos)
- #define SCB_HFSR_VECTTBL_Pos 1
- #define SCB_HFSR_VECTTBL_Msk (1UL << SCB_HFSR_VECTTBL_Pos)
- #define SCB_DFSR_EXTERNAL_Pos 4
- #define SCB_DFSR_EXTERNAL_Msk (1UL << SCB_DFSR_EXTERNAL_Pos)
- #define SCB_DFSR_VCATCH_Pos 3
- #define SCB_DFSR_VCATCH_Msk (1UL << SCB_DFSR_VCATCH_Pos)
- #define SCB_DFSR_DWTTRAP_Pos 2
- #define SCB_DFSR_DWTTRAP_Msk (1UL << SCB_DFSR_DWTTRAP_Pos)
- #define SCB_DFSR_BKPT_Pos 1
- #define SCB_DFSR_BKPT_Msk (1UL << SCB_DFSR_BKPT_Pos)
- #define SCB_DFSR_HALTED_Pos 0
- #define SCB_DFSR_HALTED_Msk (1UL << SCB_DFSR_HALTED_Pos)

## 6.92.2 Macro Definition Documentation

### 6.92.2.1 #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)

SCB AIRCR: ENDIANESS Mask

Definition at line 391 of file core_cm0plus.h.

### 6.92.2.2 #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)

SCB AIRCR: ENDIANESS Mask

Definition at line 452 of file core_cm4.h.

### 6.92.2.3 #define SCB_AIRCR_ENDIANESS_Pos 15

SCB AIRCR: ENDIANESS Position

Definition at line 390 of file core_cm0plus.h.

### 6.92.2.4 #define SCB_AIRCR_ENDIANESS_Pos 15

SCB AIRCR: ENDIANESS Position

Definition at line 451 of file core_cm4.h.

### 6.92.2.5 #define SCB_AIRCR_PRIGROUP_Msk (7UL << SCB_AIRCR_PRIGROUP_Pos)

SCB AIRCR: PRIGROUP Mask

Definition at line 455 of file core_cm4.h.

### 6.92.2.6 #define SCB_AIRCR_PRIGROUP_Pos 8

SCB AIRCR: PRIGROUP Position

Definition at line 454 of file core_cm4.h.

#### 6.92.2.7 #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)

SCB AIRCR: SYSRESETREQ Mask

Definition at line 394 of file core_cm0plus.h.

#### 6.92.2.8 #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)

SCB AIRCR: SYSRESETREQ Mask

Definition at line 458 of file core_cm4.h.

#### 6.92.2.9 #define SCB_AIRCR_SYSRESETREQ_Pos 2

SCB AIRCR: SYSRESETREQ Position

Definition at line 393 of file core_cm0plus.h.

#### 6.92.2.10 #define SCB_AIRCR_SYSRESETREQ_Pos 2

SCB AIRCR: SYSRESETREQ Position

Definition at line 457 of file core_cm4.h.

#### 6.92.2.11 #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)

SCB AIRCR: VECTCLRACTIVE Mask

Definition at line 397 of file core_cm0plus.h.

#### 6.92.2.12 #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)

SCB AIRCR: VECTCLRACTIVE Mask

Definition at line 461 of file core_cm4.h.

#### 6.92.2.13 #define SCB_AIRCR_VECTCLRACTIVE_Pos 1

SCB AIRCR: VECTCLRACTIVE Position

Definition at line 396 of file core_cm0plus.h.

#### 6.92.2.14 #define SCB_AIRCR_VECTCLRACTIVE_Pos 1

SCB AIRCR: VECTCLRACTIVE Position

Definition at line 460 of file core_cm4.h.

#### 6.92.2.15 #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)

SCB AIRCR: VECTKEY Mask

Definition at line 385 of file core_cm0plus.h.

#### 6.92.2.16 #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)

SCB AIRCR: VECTKEY Mask

Definition at line 446 of file core_cm4.h.

#### 6.92.2.17 #define SCB_AIRCR_VECTKEY_Pos 16

SCB AIRCR: VECTKEY Position

Definition at line 384 of file core_cm0plus.h.

#### 6.92.2.18 #define SCB_AIRCR_VECTKEY_Pos 16

SCB AIRCR: VECTKEY Position

Definition at line 445 of file core_cm4.h.

#### 6.92.2.19 #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)

SCB AIRCR: VECTKEYSTAT Mask

Definition at line 388 of file core_cm0plus.h.

#### 6.92.2.20 #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)

SCB AIRCR: VECTKEYSTAT Mask

Definition at line 449 of file core_cm4.h.

#### 6.92.2.21 #define SCB_AIRCR_VECTKEYSTAT_Pos 16

SCB AIRCR: VECTKEYSTAT Position

Definition at line 387 of file core_cm0plus.h.

#### 6.92.2.22 #define SCB_AIRCR_VECTKEYSTAT_Pos 16

SCB AIRCR: VECTKEYSTAT Position

Definition at line 448 of file core_cm4.h.

#### 6.92.2.23 #define SCB_AIRCR_VECTRESET_Msk (1UL << SCB_AIRCR_VECTRESET_Pos)

SCB AIRCR: VECTRESET Mask

Definition at line 464 of file core_cm4.h.

#### 6.92.2.24 #define SCB_AIRCR_VECTRESET_Pos 0

SCB AIRCR: VECTRESET Position

Definition at line 463 of file core_cm4.h.

#### 6.92.2.25 #define SCB_CCR_BFHFNMIGN_Msk (1UL << SCB_CCR_BFHFNMIGN_Pos)

SCB CCR: BFHFNMIGN Mask

Definition at line 481 of file core_cm4.h.

#### 6.92.2.26 #define SCB_CCR_BFHFNMIGN_Pos 8

SCB CCR: BFHFNMIGN Position

Definition at line 480 of file core_cm4.h.

#### 6.92.2.27 #define SCB_CCR_DIV_0_TRP_Msk (1UL << SCB_CCR_DIV_0_TRP_Pos)

SCB CCR: DIV_0_TRP Mask

Definition at line 484 of file core_cm4.h.

#### 6.92.2.28 #define SCB_CCR_DIV_0_TRP_Pos 4

SCB CCR: DIV_0_TRP Position

Definition at line 483 of file core_cm4.h.

#### 6.92.2.29 #define SCB_CCR_NONBASETHRDENA_Msk (1UL << SCB_CCR_NONBASETHRDENA_Pos)

SCB CCR: NONBASETHRDENA Mask

Definition at line 493 of file core_cm4.h.

#### 6.92.2.30 #define SCB_CCR_NONBASETHRDENA_Pos 0

SCB CCR: NONBASETHRDENA Position

Definition at line 492 of file core_cm4.h.

#### 6.92.2.31 #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)

SCB CCR: STKALIGN Mask

Definition at line 411 of file core_cm0plus.h.

#### 6.92.2.32 #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)

SCB CCR: STKALIGN Mask

Definition at line 478 of file core_cm4.h.

#### 6.92.2.33 #define SCB_CCR_STKALIGN_Pos 9

SCB CCR: STKALIGN Position

Definition at line 410 of file core_cm0plus.h.

#### 6.92.2.34 #define SCB_CCR_STKALIGN_Pos 9

SCB CCR: STKALIGN Position

Definition at line 477 of file core_cm4.h.

#### 6.92.2.35 #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)

SCB CCR: UNALIGN_TRP Mask

Definition at line 414 of file core_cm0plus.h.

#### 6.92.2.36 #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)

SCB CCR: UNALIGN_TRP Mask

Definition at line 487 of file core_cm4.h.

#### 6.92.2.37 #define SCB_CCR_UNALIGN_TRP_Pos 3

SCB CCR: UNALIGN_TRP Position

Definition at line 413 of file core_cm0plus.h.

#### 6.92.2.38 #define SCB_CCR_UNALIGN_TRP_Pos 3

SCB CCR: UNALIGN_TRP Position

Definition at line 486 of file core_cm4.h.

#### 6.92.2.39 #define SCB_CCR_USERSETMPEND_Msk (1UL << SCB_CCR_USERSETMPEND_Pos)

SCB CCR: USERSETMPEND Mask

Definition at line 490 of file core_cm4.h.

#### 6.92.2.40 #define SCB_CCR_USERSETMPEND_Pos 1

SCB CCR: USERSETMPEND Position

Definition at line 489 of file core_cm4.h.

#### 6.92.2.41 #define SCB_CFSR_BUSFAULTSR_Msk (0xFFUL << SCB_CFSR_BUSFAULTSR_Pos)

SCB CFSR: Bus Fault Status Register Mask

Definition at line 543 of file core_cm4.h.

#### 6.92.2.42 #define SCB_CFSR_BUSFAULTSR_Pos 8

SCB CFSR: Bus Fault Status Register Position

Definition at line 542 of file core_cm4.h.

#### 6.92.2.43 #define SCB_CFSR_MEMFAULTSR_Msk (0xFFUL << SCB_CFSR_MEMFAULTSR_Pos)

SCB CFSR: Memory Manage Fault Status Register Mask

Definition at line 546 of file core_cm4.h.

#### 6.92.2.44 #define SCB_CFSR_MEMFAULTSR_Pos 0

SCB CFSR: Memory Manage Fault Status Register Position

Definition at line 545 of file core_cm4.h.

#### 6.92.2.45 #define SCB_CFSR_USGFAULTSR_Msk (0xFFFFUL << SCB_CFSR_USGFAULTSR_Pos)

SCB CFSR: Usage Fault Status Register Mask

Definition at line 540 of file core_cm4.h.

#### 6.92.2.46 #define SCB_CFSR_USGFAULTSR_Pos 16

SCB CFSR: Usage Fault Status Register Position

Definition at line 539 of file core_cm4.h.

#### 6.92.2.47 #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)

SCB CPUID: ARCHITECTURE Mask

Definition at line 341 of file core_cm0plus.h.

#### 6.92.2.48 #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)

SCB CPUID: ARCHITECTURE Mask

Definition at line 401 of file core_cm4.h.

#### 6.92.2.49 #define SCB_CPUID_ARCHITECTURE_Pos 16

SCB CPUID: ARCHITECTURE Position

Definition at line 340 of file core_cm0plus.h.

#### 6.92.2.50 #define SCB_CPUID_ARCHITECTURE_Pos 16

SCB CPUID: ARCHITECTURE Position

Definition at line 400 of file core_cm4.h.

#### 6.92.2.51 #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)

SCB CPUID: IMPLEMENTER Mask

Definition at line 335 of file core_cm0plus.h.

#### 6.92.2.52 #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)

SCB CPUID: IMPLEMENTER Mask

Definition at line 395 of file core_cm4.h.

#### 6.92.2.53 #define SCB_CPUID_IMPLEMENTER_Pos 24

SCB CPUID: IMPLEMENTER Position

Definition at line 334 of file core_cm0plus.h.

#### 6.92.2.54 #define SCB_CPUID_IMPLEMENTER_Pos 24

SCB CPUID: IMPLEMENTER Position

Definition at line 394 of file core_cm4.h.

#### 6.92.2.55 #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)

SCB CPUID: PARTNO Mask

Definition at line 344 of file core_cm0plus.h.

#### 6.92.2.56 #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)

SCB CPUID: PARTNO Mask

Definition at line 404 of file core_cm4.h.

#### 6.92.2.57 #define SCB_CPUID_PARTNO_Pos 4

SCB CPUID: PARTNO Position

Definition at line 343 of file core_cm0plus.h.

#### 6.92.2.58 #define SCB_CPUID_PARTNO_Pos 4

SCB CPUID: PARTNO Position

Definition at line 403 of file core_cm4.h.

#### 6.92.2.59 #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)

SCB CPUID: REVISION Mask

Definition at line 347 of file core_cm0plus.h.

#### 6.92.2.60 #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)

SCB CPUID: REVISION Mask

Definition at line 407 of file core_cm4.h.

#### 6.92.2.61 #define SCB_CPUID_REVISION_Pos 0

SCB CPUID: REVISION Position

Definition at line 346 of file core_cm0plus.h.

#### 6.92.2.62 #define SCB_CPUID_REVISION_Pos 0

SCB CPUID: REVISION Position

Definition at line 406 of file core_cm4.h.

#### 6.92.2.63 #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)

SCB CPUID: VARIANT Mask

Definition at line 338 of file core_cm0plus.h.

#### 6.92.2.64 #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)

SCB CPUID: VARIANT Mask

Definition at line 398 of file core_cm4.h.

#### 6.92.2.65 #define SCB_CPUID_VARIANT_Pos 20

SCB CPUID: VARIANT Position

Definition at line 337 of file core_cm0plus.h.

#### 6.92.2.66 #define SCB_CPUID_VARIANT_Pos 20

SCB CPUID: VARIANT Position

Definition at line 397 of file core_cm4.h.

#### 6.92.2.67 #define SCB_DFSR_BKPT_Msk (1UL << SCB_DFSR_BKPT_Pos)

SCB DFSR: BKPT Mask

Definition at line 569 of file core_cm4.h.

#### 6.92.2.68 #define SCB_DFSR_BKPT_Pos 1

SCB DFSR: BKPT Position

Definition at line 568 of file core_cm4.h.

#### 6.92.2.69 #define SCB_DFSR_DWTTRAP_Msk (1UL << SCB_DFSR_DWTTRAP_Pos)

SCB DFSR: DWTTRAP Mask

Definition at line 566 of file core_cm4.h.

#### 6.92.2.70 #define SCB_DFSR_DWTTRAP_Pos 2

SCB DFSR: DWTTRAP Position

Definition at line 565 of file core_cm4.h.

#### 6.92.2.71 #define SCB_DFSR_EXTERNAL_Msk (1UL << SCB_DFSR_EXTERNAL_Pos)

SCB DFSR: EXTERNAL Mask

Definition at line 560 of file core_cm4.h.

#### 6.92.2.72 #define SCB_DFSR_EXTERNAL_Pos 4

SCB DFSR: EXTERNAL Position

Definition at line 559 of file core_cm4.h.

#### 6.92.2.73 #define SCB_DFSR_HALTED_Msk (1UL << SCB_DFSR_HALTED_Pos)

SCB DFSR: HALTED Mask

Definition at line 572 of file core_cm4.h.

#### 6.92.2.74 #define SCB_DFSR_HALTED_Pos 0

SCB DFSR: HALTED Position

Definition at line 571 of file core_cm4.h.

#### 6.92.2.75 #define SCB_DFSR_VCATCH_Msk (1UL << SCB_DFSR_VCATCH_Pos)

SCB DFSR: VCATCH Mask

Definition at line 563 of file core_cm4.h.

#### 6.92.2.76 #define SCB_DFSR_VCATCH_Pos 3

SCB DFSR: VCATCH Position

Definition at line 562 of file core_cm4.h.

#### 6.92.2.77 #define SCB_HFSR_DEBUGEVT_Msk (1UL << SCB_HFSR_DEBUGEVT_Pos)

SCB HFSR: DEBUGEVT Mask

Definition at line 550 of file core_cm4.h.

#### 6.92.2.78 #define SCB_HFSR_DEBUGEVT_Pos 31

SCB HFSR: DEBUGEVT Position

Definition at line 549 of file core_cm4.h.

#### 6.92.2.79 #define SCB_HFSR_FORCED_Msk (1UL << SCB_HFSR_FORCED_Pos)

SCB HFSR: FORCED Mask

Definition at line 553 of file core_cm4.h.

#### 6.92.2.80 #define SCB_HFSR_FORCED_Pos 30

SCB HFSR: FORCED Position

Definition at line 552 of file core_cm4.h.

#### 6.92.2.81 #define SCB_HFSR_VECTTBL_Msk (1UL << SCB_HFSR_VECTTBL_Pos)

SCB HFSR: VECTTBL Mask

Definition at line 556 of file core_cm4.h.

#### 6.92.2.82 #define SCB_HFSR_VECTTBL_Pos 1

SCB HFSR: VECTTBL Position

Definition at line 555 of file core_cm4.h.

#### 6.92.2.83 #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)

SCB ICSR: ISRPENDING Mask

Definition at line 369 of file core_cm0plus.h.

#### 6.92.2.84 #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)

SCB ICSR: ISRPENDING Mask

Definition at line 429 of file core_cm4.h.

#### 6.92.2.85 #define SCB_ICSR_ISRPENDING_Pos 22

SCB ICSR: ISRPENDING Position

Definition at line 368 of file core_cm0plus.h.

#### 6.92.2.86 #define SCB_ICSR_ISRPENDING_Pos 22

SCB ICSR: ISRPENDING Position

Definition at line 428 of file core_cm4.h.

#### 6.92.2.87 #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)

SCB ICSR: ISRPREEMPT Mask

Definition at line 366 of file core_cm0plus.h.

#### 6.92.2.88 #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)

SCB ICSR: ISRPREEMPT Mask

Definition at line 426 of file core_cm4.h.

#### 6.92.2.89 #define SCB_ICSR_ISRPREEMPT_Pos 23

SCB ICSR: ISRPREEMPT Position

Definition at line 365 of file core_cm0plus.h.

#### 6.92.2.90 #define SCB_ICSR_ISRPREEMPT_Pos 23

SCB ICSR: ISRPREEMPT Position

Definition at line 425 of file core_cm4.h.

#### 6.92.2.91 #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)

SCB ICSR: NMIPENDSET Mask

Definition at line 351 of file core_cm0plus.h.

#### 6.92.2.92 #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)

SCB ICSR: NMIPENDSET Mask

Definition at line 411 of file core_cm4.h.

#### 6.92.2.93 #define SCB_ICSR_NMIPENDSET_Pos 31

SCB ICSR: NMIPENDSET Position

Definition at line 350 of file core_cm0plus.h.

#### 6.92.2.94 #define SCB_ICSR_NMIPENDSET_Pos 31

SCB ICSR: NMIPENDSET Position

Definition at line 410 of file core_cm4.h.

#### 6.92.2.95 #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)

SCB ICSR: PENDSTCLR Mask

Definition at line 363 of file core_cm0plus.h.

#### 6.92.2.96 #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)

SCB ICSR: PENDSTCLR Mask

Definition at line 423 of file core_cm4.h.

#### 6.92.2.97 #define SCB_ICSR_PENDSTCLR_Pos 25

SCB ICSR: PENDSTCLR Position

Definition at line 362 of file core_cm0plus.h.

#### 6.92.2.98 #define SCB_ICSR_PENDSTCLR_Pos 25

SCB ICSR: PENDSTCLR Position

Definition at line 422 of file core_cm4.h.

#### 6.92.2.99 #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)

SCB ICSR: PENDSTSET Mask

Definition at line 360 of file core_cm0plus.h.

#### 6.92.2.100 #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)

SCB ICSR: PENDSTSET Mask

Definition at line 420 of file core_cm4.h.

#### 6.92.2.101 #define SCB_ICSR_PENDSTSET_Pos 26

SCB ICSR: PENDSTSET Position

Definition at line 359 of file core_cm0plus.h.

#### 6.92.2.102 #define SCB_ICSR_PENDSTSET_Pos 26

SCB ICSR: PENDSTSET Position

Definition at line 419 of file core_cm4.h.

#### 6.92.2.103 #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)

SCB ICSR: PENDSVCLR Mask

Definition at line 357 of file core_cm0plus.h.

#### 6.92.2.104 #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)

SCB ICSR: PENDSVCLR Mask

Definition at line 417 of file core_cm4.h.

#### 6.92.2.105 #define SCB_ICSR_PENDSVCLR_Pos 27

SCB ICSR: PENDSVCLR Position

Definition at line 356 of file core_cm0plus.h.

#### 6.92.2.106 #define SCB_ICSR_PENDSVCLR_Pos 27

SCB ICSR: PENDSVCLR Position

Definition at line 416 of file core_cm4.h.

#### 6.92.2.107 #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)

SCB ICSR: PENDSVSET Mask

Definition at line 354 of file core_cm0plus.h.

#### 6.92.2.108 #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)

SCB ICSR: PENDSVSET Mask

Definition at line 414 of file core_cm4.h.

#### 6.92.2.109 #define SCB_ICSR_PENDSVSET_Pos 28

SCB ICSR: PENDSVSET Position

Definition at line 353 of file core_cm0plus.h.

#### 6.92.2.110 #define SCB_ICSR_PENDSVSET_Pos 28

SCB ICSR: PENDSVSET Position

Definition at line 413 of file core_cm4.h.

#### 6.92.2.111 #define SCB_ICSR_RETTOBASE_Msk (1UL << SCB_ICSR_RETTOBASE_Pos)

SCB ICSR: RETTOBASE Mask

Definition at line 435 of file core_cm4.h.

#### 6.92.2.112 #define SCB_ICSR_RETTOBASE_Pos 11

SCB ICSR: RETTOBASE Position

Definition at line 434 of file core_cm4.h.

#### 6.92.2.113 #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)

SCB ICSR: VECTACTIVE Mask

Definition at line 375 of file core_cm0plus.h.

#### 6.92.2.114 #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)

SCB ICSR: VECTACTIVE Mask

Definition at line 438 of file core_cm4.h.

#### 6.92.2.115 #define SCB_ICSR_VECTACTIVE_Pos 0

SCB ICSR: VECTACTIVE Position

Definition at line 374 of file core_cm0plus.h.

#### 6.92.2.116 #define SCB_ICSR_VECTACTIVE_Pos 0

SCB ICSR: VECTACTIVE Position

Definition at line 437 of file core_cm4.h.

#### 6.92.2.117 #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)

SCB ICSR: VECTPENDING Mask

Definition at line 372 of file core_cm0plus.h.

#### 6.92.2.118 #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)

SCB ICSR: VECTPENDING Mask

Definition at line 432 of file core_cm4.h.

#### 6.92.2.119 #define SCB_ICSR_VECTPENDING_Pos 12

SCB ICSR: VECTPENDING Position

Definition at line 371 of file core_cm0plus.h.

#### 6.92.2.120 #define SCB_ICSR_VECTPENDING_Pos 12

SCB ICSR: VECTPENDING Position

Definition at line 431 of file core_cm4.h.

#### 6.92.2.121 #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)

SCB SCR: SEVONPEND Mask

Definition at line 401 of file core_cm0plus.h.

#### 6.92.2.122 #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)

SCB SCR: SEVONPEND Mask

Definition at line 468 of file core_cm4.h.

#### 6.92.2.123 #define SCB_SCR_SEVONPEND_Pos 4

SCB SCR: SEVONPEND Position

Definition at line 400 of file core_cm0plus.h.

#### 6.92.2.124 #define SCB_SCR_SEVONPEND_Pos 4

SCB SCR: SEVONPEND Position

Definition at line 467 of file core_cm4.h.

#### 6.92.2.125 #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)

SCB SCR: SLEEPDEEP Mask

Definition at line 404 of file core_cm0plus.h.

#### 6.92.2.126 #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)

SCB SCR: SLEEPDEEP Mask

Definition at line 471 of file core_cm4.h.

#### 6.92.2.127 #define SCB_SCR_SLEEPDEEP_Pos 2

SCB SCR: SLEEPDEEP Position

Definition at line 403 of file core_cm0plus.h.

#### 6.92.2.128 #define SCB_SCR_SLEEPDEEP_Pos 2

SCB SCR: SLEEPDEEP Position

Definition at line 470 of file core_cm4.h.

#### 6.92.2.129 #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)

SCB SCR: SLEEPONEXIT Mask

Definition at line 407 of file core_cm0plus.h.

#### 6.92.2.130 #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)

SCB SCR: SLEEPONEXIT Mask

Definition at line 474 of file core_cm4.h.

#### 6.92.2.131 #define SCB_SCR_SLEEPONEXIT_Pos 1

SCB SCR: SLEEPONEXIT Position

Definition at line 406 of file core_cm0plus.h.

#### 6.92.2.132 #define SCB_SCR_SLEEPONEXIT_Pos 1

SCB SCR: SLEEPONEXIT Position

Definition at line 473 of file core_cm4.h.

#### 6.92.2.133 #define SCB_SHCSR_BUSFAULTACT_Msk (1UL << SCB_SHCSR_BUSFAULTACT_Pos)

SCB SHCSR: BUSFAULTACT Mask

Definition at line 533 of file core_cm4.h.

#### 6.92.2.134 #define SCB_SHCSR_BUSFAULTACT_Pos 1

SCB SHCSR: BUSFAULTACT Position

Definition at line 532 of file core_cm4.h.

#### 6.92.2.135 #define SCB_SHCSR_BUSFAULTENA_Msk (1UL << SCB_SHCSR_BUSFAULTENA_Pos)

SCB SHCSR: BUSFAULTENA Mask

Definition at line 500 of file core_cm4.h.

#### 6.92.2.136 #define SCB_SHCSR_BUSFAULTENA_Pos 17

SCB SHCSR: BUSFAULTENA Position

Definition at line 499 of file core_cm4.h.

#### 6.92.2.137 #define SCB_SHCSR_BUSFAULTPENDED_Msk (1UL << SCB_SHCSR_BUSFAULTPENDED_Pos)

SCB SHCSR: BUSFAULTPENDED Mask

Definition at line 509 of file core_cm4.h.

#### 6.92.2.138 #define SCB_SHCSR_BUSFAULTPENDED_Pos 14

SCB SHCSR: BUSFAULTPENDED Position

Definition at line 508 of file core_cm4.h.

#### 6.92.2.139 #define SCB_SHCSR_MEMFAULTACT_Msk (1UL << SCB_SHCSR_MEMFAULTACT_Pos)

SCB SHCSR: MEMFAULTACT Mask

Definition at line 536 of file core_cm4.h.

#### 6.92.2.140 #define SCB_SHCSR_MEMFAULTACT_Pos 0

SCB SHCSR: MEMFAULTACT Position

Definition at line 535 of file core_cm4.h.

#### 6.92.2.141 #define SCB_SHCSR_MEMFAULTENA_Msk (1UL << SCB_SHCSR_MEMFAULTENA_Pos)

SCB SHCSR: MEMFAULTENA Mask

Definition at line 503 of file core_cm4.h.

#### 6.92.2.142 #define SCB_SHCSR_MEMFAULTENA_Pos 16

SCB SHCSR: MEMFAULTENA Position

Definition at line 502 of file core_cm4.h.

#### 6.92.2.143 #define SCB_SHCSR_MEMFAULTPENDED_Msk (1UL << SCB_SHCSR_MEMFAULTPENDED_Pos)

SCB SHCSR: MEMFAULTPENDED Mask

Definition at line 512 of file core_cm4.h.

#### 6.92.2.144 #define SCB_SHCSR_MEMFAULTPENDED_Pos 13

SCB SHCSR: MEMFAULTPENDED Position

Definition at line 511 of file core_cm4.h.

#### 6.92.2.145 #define SCB_SHCSR_MONITORACT_Msk (1UL << SCB_SHCSR_MONITORACT_Pos)

SCB SHCSR: MONITORACT Mask

Definition at line 524 of file core_cm4.h.

#### 6.92.2.146 #define SCB_SHCSR_MONITORACT_Pos 8

SCB SHCSR: MONITORACT Position

Definition at line 523 of file core_cm4.h.

#### 6.92.2.147 #define SCB_SHCSR_PENDSVACT_Msk (1UL << SCB_SHCSR_PENDSVACT_Pos)

SCB SHCSR: PENDSVACT Mask

Definition at line 521 of file core_cm4.h.

#### 6.92.2.148 #define SCB_SHCSR_PENDSVACT_Pos 10

SCB SHCSR: PENDSVACT Position

Definition at line 520 of file core_cm4.h.

#### 6.92.2.149 #define SCB_SHCSR_SVCALLACT_Msk (1UL << SCB_SHCSR_SVCALLACT_Pos)

SCB SHCSR: SVCALLACT Mask

Definition at line 527 of file core_cm4.h.

#### 6.92.2.150 #define SCB_SHCSR_SVCALLACT_Pos 7

SCB SHCSR: SVCALLACT Position

Definition at line 526 of file core_cm4.h.

#### 6.92.2.151 #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)

SCB SHCSR: SVCALLPENDED Mask

Definition at line 418 of file core_cm0plus.h.

#### 6.92.2.152 #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)

SCB SHCSR: SVCALLPENDED Mask

Definition at line 506 of file core_cm4.h.

#### 6.92.2.153 #define SCB_SHCSR_SVCALLPENDED_Pos 15

SCB SHCSR: SVCALLPENDED Position

Definition at line 417 of file core_cm0plus.h.

#### 6.92.2.154 #define SCB_SHCSR_SVCALLPENDED_Pos 15

SCB SHCSR: SVCALLPENDED Position

Definition at line 505 of file core_cm4.h.

#### 6.92.2.155 #define SCB_SHCSR_SYSTICKACT_Msk (1UL << SCB_SHCSR_SYSTICKACT_Pos)

SCB SHCSR: SYSTICKACT Mask

Definition at line 518 of file core_cm4.h.

#### 6.92.2.156 #define SCB_SHCSR_SYSTICKACT_Pos 11

SCB SHCSR: SYSTICKACT Position

Definition at line 517 of file core_cm4.h.

#### 6.92.2.157 #define SCB_SHCSR_USGFAULTACT_Msk (1UL << SCB_SHCSR_USGFAULTACT_Pos)

SCB SHCSR: USGFAULTACT Mask

Definition at line 530 of file core_cm4.h.

#### 6.92.2.158 #define SCB_SHCSR_USGFAULTACT_Pos 3

SCB SHCSR: USGFAULTACT Position

Definition at line 529 of file core_cm4.h.

#### 6.92.2.159 #define SCB_SHCSR_USGFAULTENA_Msk (1UL << SCB_SHCSR_USGFAULTENA_Pos)

SCB SHCSR: USGFAULTENA Mask

Definition at line 497 of file core_cm4.h.

#### 6.92.2.160 #define SCB_SHCSR_USGFAULTENA_Pos 18

SCB SHCSR: USGFAULTENA Position

Definition at line 496 of file core_cm4.h.

#### 6.92.2.161 #define SCB_SHCSR_USGFAULTPENDED_Msk (1UL << SCB_SHCSR_USGFAULTPENDED_Pos)

SCB SHCSR: USGFAULTPENDED Mask

Definition at line 515 of file core_cm4.h.

#### 6.92.2.162 #define SCB_SHCSR_USGFAULTPENDED_Pos 12

SCB SHCSR: USGFAULTPENDED Position

Definition at line 514 of file core_cm4.h.

#### 6.92.2.163 #define SCB_VTOR_TBLOFF_Msk (0x1FFFFFFUL << SCB_VTOR_TBLOFF_Pos)

SCB VTOR: TBLOFF Mask

Definition at line 442 of file core_cm4.h.

#### 6.92.2.164 #define SCB_VTOR_TBLOFF_Pos 7

SCB VTOR: TBLOFF Position

Definition at line 441 of file core_cm4.h.

## 6.93 System Controls not in SCB (SCnSCB)

### 6.93.1 Detailed Description

Type definitions for the System Control and ID Register not in the SCB.

#### Data Structures

- struct SCnSCB_Type

  *Structure type to access the System Control and ID Register not in the SCB.*

#### Macros

- #define SCnSCB_ICTR_INTLINESNUM_Pos 0
- #define SCnSCB_ICTR_INTLINESNUM_Msk (0xFUL << SCnSCB_ICTR_INTLINESNUM_Pos)
- #define SCnSCB_ACTLR_DISOOFP_Pos 9
- #define SCnSCB_ACTLR_DISOOFP_Msk (1UL << SCnSCB_ACTLR_DISOOFP_Pos)
- #define SCnSCB_ACTLR_DISFPCA_Pos 8
- #define SCnSCB_ACTLR_DISFPCA_Msk (1UL << SCnSCB_ACTLR_DISFPCA_Pos)
- #define SCnSCB_ACTLR_DISFOLD_Pos 2
- #define SCnSCB_ACTLR_DISFOLD_Msk (1UL << SCnSCB_ACTLR_DISFOLD_Pos)
- #define SCnSCB_ACTLR_DISDEFWBUF_Pos 1
- #define SCnSCB_ACTLR_DISDEFWBUF_Msk (1UL << SCnSCB_ACTLR_DISDEFWBUF_Pos)
- #define SCnSCB_ACTLR_DISMCYCINT_Pos 0
- #define SCnSCB_ACTLR_DISMCYCINT_Msk (1UL << SCnSCB_ACTLR_DISMCYCINT_Pos)

### 6.93.2 Macro Definition Documentation

#### 6.93.2.1 #define SCnSCB_ACTLR_DISDEFWBUF_Msk (1UL << SCnSCB_ACTLR_DISDEFWBUF_Pos)

ACTLR: DISDEFWBUF Mask

Definition at line 607 of file core_cm4.h.

#### 6.93.2.2 #define SCnSCB_ACTLR_DISDEFWBUF_Pos 1

ACTLR: DISDEFWBUF Position

Definition at line 606 of file core_cm4.h.

#### 6.93.2.3 #define SCnSCB_ACTLR_DISFOLD_Msk (1UL << SCnSCB_ACTLR_DISFOLD_Pos)

ACTLR: DISFOLD Mask

Definition at line 604 of file core_cm4.h.

#### 6.93.2.4 #define SCnSCB_ACTLR_DISFOLD_Pos 2

ACTLR: DISFOLD Position

Definition at line 603 of file core_cm4.h.

#### 6.93.2.5 #define SCnSCB_ACTLR_DISFPCA_Msk (1UL << SCnSCB_ACTLR_DISFPCA_Pos)

ACTLR: DISFPCA Mask

Definition at line 601 of file core_cm4.h.

#### 6.93.2.6 #define SCnSCB_ACTLR_DISFPCA_Pos 8

ACTLR: DISFPCA Position

Definition at line 600 of file core_cm4.h.

#### 6.93.2.7 #define SCnSCB_ACTLR_DISMCYCINT_Msk (1UL << SCnSCB_ACTLR_DISMCYCINT_Pos)

ACTLR: DISMCYCINT Mask

Definition at line 610 of file core_cm4.h.

#### 6.93.2.8 #define SCnSCB_ACTLR_DISMCYCINT_Pos 0

ACTLR: DISMCYCINT Position

Definition at line 609 of file core_cm4.h.

#### 6.93.2.9 #define SCnSCB_ACTLR_DISOOFP_Msk (1UL << SCnSCB_ACTLR_DISOOFP_Pos)

ACTLR: DISOOFP Mask

Definition at line 598 of file core_cm4.h.

#### 6.93.2.10 #define SCnSCB_ACTLR_DISOOFP_Pos 9

ACTLR: DISOOFP Position

Definition at line 597 of file core_cm4.h.

#### 6.93.2.11 #define SCnSCB_ICTR_INTLINESNUM_Msk (0xFUL << SCnSCB_ICTR_INTLINESNUM_Pos)

ICTR: INTLINESNUM Mask

Definition at line 594 of file core_cm4.h.

#### 6.93.2.12 #define SCnSCB_ICTR_INTLINESNUM_Pos 0

ICTR: INTLINESNUM Position

Definition at line 593 of file core_cm4.h.

## 6.94 System Tick Timer (SysTick)

### 6.94.1 Detailed Description

Type definitions for the System Timer Registers.

#### Data Structures

- struct SysTick_Type

  *Structure type to access the System Timer (SysTick).*

#### Macros

- #define SysTick_CTRL_COUNTFLAG_Pos 16
- #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)
- #define SysTick_CTRL_CLKSOURCE_Pos 2
- #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)
- #define SysTick_CTRL_TICKINT_Pos 1
- #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)
- #define SysTick_CTRL_ENABLE_Pos 0
- #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)
- #define SysTick_LOAD_RELOAD_Pos 0
- #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)
- #define SysTick_VAL_CURRENT_Pos 0
- #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define SysTick_CALIB_NOREF_Pos 31
- #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)
- #define SysTick_CALIB_SKEW_Pos 30
- #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)
- #define SysTick_CALIB_TENMS_Pos 0
- #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)

- #define SysTick_CTRL_COUNTFLAG_Pos 16
- #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)
- #define SysTick_CTRL_CLKSOURCE_Pos 2
- #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)
- #define SysTick_CTRL_TICKINT_Pos 1
- #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)
- #define SysTick_CTRL_ENABLE_Pos 0
- #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)
- #define SysTick_LOAD_RELOAD_Pos 0
- #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)
- #define SysTick_VAL_CURRENT_Pos 0
- #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define SysTick_CALIB_NOREF_Pos 31
- #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)
- #define SysTick_CALIB_SKEW_Pos 30
- #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)
- #define SysTick_CALIB_TENMS_Pos 0
- #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)

## 6.94.2 Macro Definition Documentation

### 6.94.2.1 #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)

SysTick CALIB: NOREF Mask

Definition at line 462 of file core_cm0plus.h.

### 6.94.2.2 #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)

SysTick CALIB: NOREF Mask

Definition at line 654 of file core_cm4.h.

### 6.94.2.3 #define SysTick_CALIB_NOREF_Pos 31

SysTick CALIB: NOREF Position

Definition at line 461 of file core_cm0plus.h.

### 6.94.2.4 #define SysTick_CALIB_NOREF_Pos 31

SysTick CALIB: NOREF Position

Definition at line 653 of file core_cm4.h.

### 6.94.2.5 #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)

SysTick CALIB: SKEW Mask

Definition at line 465 of file core_cm0plus.h.

### 6.94.2.6 #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)

SysTick CALIB: SKEW Mask

Definition at line 657 of file core_cm4.h.

### 6.94.2.7 #define SysTick_CALIB_SKEW_Pos 30

SysTick CALIB: SKEW Position

Definition at line 464 of file core_cm0plus.h.

### 6.94.2.8 #define SysTick_CALIB_SKEW_Pos 30

SysTick CALIB: SKEW Position

Definition at line 656 of file core_cm4.h.

### 6.94.2.9 #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)

SysTick CALIB: TENMS Mask

Definition at line 468 of file core_cm0plus.h.

#### 6.94.2.10 #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)

SysTick CALIB: TENMS Mask

Definition at line 660 of file core_cm4.h.

#### 6.94.2.11 #define SysTick_CALIB_TENMS_Pos 0

SysTick CALIB: TENMS Position

Definition at line 467 of file core_cm0plus.h.

#### 6.94.2.12 #define SysTick_CALIB_TENMS_Pos 0

SysTick CALIB: TENMS Position

Definition at line 659 of file core_cm4.h.

#### 6.94.2.13 #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)

SysTick CTRL: CLKSOURCE Mask

Definition at line 444 of file core_cm0plus.h.

#### 6.94.2.14 #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)

SysTick CTRL: CLKSOURCE Mask

Definition at line 636 of file core_cm4.h.

#### 6.94.2.15 #define SysTick_CTRL_CLKSOURCE_Pos 2

SysTick CTRL: CLKSOURCE Position

Definition at line 443 of file core_cm0plus.h.

#### 6.94.2.16 #define SysTick_CTRL_CLKSOURCE_Pos 2

SysTick CTRL: CLKSOURCE Position

Definition at line 635 of file core_cm4.h.

#### 6.94.2.17 #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)

SysTick CTRL: COUNTFLAG Mask

Definition at line 441 of file core_cm0plus.h.

#### 6.94.2.18 #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)

SysTick CTRL: COUNTFLAG Mask

Definition at line 633 of file core_cm4.h.

#### 6.94.2.19 #define SysTick_CTRL_COUNTFLAG_Pos 16

SysTick CTRL: COUNTFLAG Position

Definition at line 440 of file core_cm0plus.h.

#### 6.94.2.20 #define SysTick_CTRL_COUNTFLAG_Pos 16

SysTick CTRL: COUNTFLAG Position

Definition at line 632 of file core_cm4.h.

#### 6.94.2.21 #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)

SysTick CTRL: ENABLE Mask

Definition at line 450 of file core_cm0plus.h.

#### 6.94.2.22 #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)

SysTick CTRL: ENABLE Mask

Definition at line 642 of file core_cm4.h.

#### 6.94.2.23 #define SysTick_CTRL_ENABLE_Pos 0

SysTick CTRL: ENABLE Position

Definition at line 449 of file core_cm0plus.h.

#### 6.94.2.24 #define SysTick_CTRL_ENABLE_Pos 0

SysTick CTRL: ENABLE Position

Definition at line 641 of file core_cm4.h.

#### 6.94.2.25 #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)

SysTick CTRL: TICKINT Mask

Definition at line 447 of file core_cm0plus.h.

#### 6.94.2.26 #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)

SysTick CTRL: TICKINT Mask

Definition at line 639 of file core_cm4.h.

#### 6.94.2.27 #define SysTick_CTRL_TICKINT_Pos 1

SysTick CTRL: TICKINT Position

Definition at line 446 of file core_cm0plus.h.

#### 6.94.2.28 #define SysTick_CTRL_TICKINT_Pos 1

SysTick CTRL: TICKINT Position

Definition at line 638 of file core_cm4.h.

#### 6.94.2.29 #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)

SysTick LOAD: RELOAD Mask

Definition at line 454 of file core_cm0plus.h.

#### 6.94.2.30 #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)

SysTick LOAD: RELOAD Mask

Definition at line 646 of file core_cm4.h.

#### 6.94.2.31 #define SysTick_LOAD_RELOAD_Pos 0

SysTick LOAD: RELOAD Position

Definition at line 453 of file core_cm0plus.h.

#### 6.94.2.32 #define SysTick_LOAD_RELOAD_Pos 0

SysTick LOAD: RELOAD Position

Definition at line 645 of file core_cm4.h.

#### 6.94.2.33 #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)

SysTick VAL: CURRENT Mask

Definition at line 458 of file core_cm0plus.h.

#### 6.94.2.34 #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)

SysTick VAL: CURRENT Mask

Definition at line 650 of file core_cm4.h.

#### 6.94.2.35 #define SysTick_VAL_CURRENT_Pos 0

SysTick VAL: CURRENT Position

Definition at line 457 of file core_cm0plus.h.

#### 6.94.2.36 #define SysTick_VAL_CURRENT_Pos 0

SysTick VAL: CURRENT Position

Definition at line 649 of file core_cm4.h.

## 6.95 Trace Port Interface (TPI)

### 6.95.1 Detailed Description

Type definitions for the Trace Port Interface (TPI)

#### Data Structures

- struct TPI_Type

  *Structure type to access the Trace Port Interface Register (TPI).*

#### Macros

- #define TPI_ACPR_PRESCALER_Pos 0
- #define TPI_ACPR_PRESCALER_Msk (0x1FFFUL << TPI_ACPR_PRESCALER_Pos)
- #define TPI_SPPR_TXMODE_Pos 0
- #define TPI_SPPR_TXMODE_Msk (0x3UL << TPI_SPPR_TXMODE_Pos)
- #define TPI_FFSR_FtNonStop_Pos 3
- #define TPI_FFSR_FtNonStop_Msk (0x1UL << TPI_FFSR_FtNonStop_Pos)
- #define TPI_FFSR_TCPresent_Pos 2
- #define TPI_FFSR_TCPresent_Msk (0x1UL << TPI_FFSR_TCPresent_Pos)
- #define TPI_FFSR_FtStopped_Pos 1
- #define TPI_FFSR_FtStopped_Msk (0x1UL << TPI_FFSR_FtStopped_Pos)
- #define TPI_FFSR_FlInProg_Pos 0
- #define TPI_FFSR_FlInProg_Msk (0x1UL << TPI_FFSR_FlInProg_Pos)
- #define TPI_FFCR_TrigIn_Pos 8
- #define TPI_FFCR_TrigIn_Msk (0x1UL << TPI_FFCR_TrigIn_Pos)
- #define TPI_FFCR_EnFCont_Pos 1
- #define TPI_FFCR_EnFCont_Msk (0x1UL << TPI_FFCR_EnFCont_Pos)
- #define TPI_TRIGGER_TRIGGER_Pos 0
- #define TPI_TRIGGER_TRIGGER_Msk (0x1UL << TPI_TRIGGER_TRIGGER_Pos)
- #define TPI_FIFO0_ITM_ATVALID_Pos 29
- #define TPI_FIFO0_ITM_ATVALID_Msk (0x3UL << TPI_FIFO0_ITM_ATVALID_Pos)
- #define TPI_FIFO0_ITM_bytecount_Pos 27
- #define TPI_FIFO0_ITM_bytecount_Msk (0x3UL << TPI_FIFO0_ITM_bytecount_Pos)
- #define TPI_FIFO0_ETM_ATVALID_Pos 26
- #define TPI_FIFO0_ETM_ATVALID_Msk (0x3UL << TPI_FIFO0_ETM_ATVALID_Pos)
- #define TPI_FIFO0_ETM_bytecount_Pos 24
- #define TPI_FIFO0_ETM_bytecount_Msk (0x3UL << TPI_FIFO0_ETM_bytecount_Pos)
- #define TPI_FIFO0_ETM2_Pos 16
- #define TPI_FIFO0_ETM2_Msk (0xFFUL << TPI_FIFO0_ETM2_Pos)
- #define TPI_FIFO0_ETM1_Pos 8
- #define TPI_FIFO0_ETM1_Msk (0xFFUL << TPI_FIFO0_ETM1_Pos)
- #define TPI_FIFO0_ETM0_Pos 0
- #define TPI_FIFO0_ETM0_Msk (0xFFUL << TPI_FIFO0_ETM0_Pos)
- #define TPI_ITATBCTR2_ATREADY_Pos 0
- #define TPI_ITATBCTR2_ATREADY_Msk (0x1UL << TPI_ITATBCTR2_ATREADY_Pos)
- #define TPI_FIFO1_ITM_ATVALID_Pos 29
- #define TPI_FIFO1_ITM_ATVALID_Msk (0x3UL << TPI_FIFO1_ITM_ATVALID_Pos)
- #define TPI_FIFO1_ITM_bytecount_Pos 27
- #define TPI_FIFO1_ITM_bytecount_Msk (0x3UL << TPI_FIFO1_ITM_bytecount_Pos)
- #define TPI_FIFO1_ETM_ATVALID_Pos 26
- #define TPI_FIFO1_ETM_ATVALID_Msk (0x3UL << TPI_FIFO1_ETM_ATVALID_Pos)

- #define TPI_FIFO1_ETM_bytecount_Pos 24
- #define TPI_FIFO1_ETM_bytecount_Msk (0x3UL << TPI_FIFO1_ETM_bytecount_Pos)
- #define TPI_FIFO1_ITM2_Pos 16
- #define TPI_FIFO1_ITM2_Msk (0xFFUL << TPI_FIFO1_ITM2_Pos)
- #define TPI_FIFO1_ITM1_Pos 8
- #define TPI_FIFO1_ITM1_Msk (0xFFUL << TPI_FIFO1_ITM1_Pos)
- #define TPI_FIFO1_ITM0_Pos 0
- #define TPI_FIFO1_ITM0_Msk (0xFFUL << TPI_FIFO1_ITM0_Pos)
- #define TPI_ITATBCTR0_ATREADY_Pos 0
- #define TPI_ITATBCTR0_ATREADY_Msk (0x1UL << TPI_ITATBCTR0_ATREADY_Pos)
- #define TPI_ITCTRL_Mode_Pos 0
- #define TPI_ITCTRL_Mode_Msk (0x1UL << TPI_ITCTRL_Mode_Pos)
- #define TPI_DEVID_NRZVALID_Pos 11
- #define TPI_DEVID_NRZVALID_Msk (0x1UL << TPI_DEVID_NRZVALID_Pos)
- #define TPI_DEVID_MANCVALID_Pos 10
- #define TPI_DEVID_MANCVALID_Msk (0x1UL << TPI_DEVID_MANCVALID_Pos)
- #define TPI_DEVID_PTINVALID_Pos 9
- #define TPI_DEVID_PTINVALID_Msk (0x1UL << TPI_DEVID_PTINVALID_Pos)
- #define TPI_DEVID_MinBufSz_Pos 6
- #define TPI_DEVID_MinBufSz_Msk (0x7UL << TPI_DEVID_MinBufSz_Pos)
- #define TPI_DEVID_AsynClkIn_Pos 5
- #define TPI_DEVID_AsynClkIn_Msk (0x1UL << TPI_DEVID_AsynClkIn_Pos)
- #define TPI_DEVID_NrTraceInput_Pos 0
- #define TPI_DEVID_NrTraceInput_Msk (0x1FUL << TPI_DEVID_NrTraceInput_Pos)
- #define TPI_DEVTYPE_SubType_Pos 0
- #define TPI_DEVTYPE_SubType_Msk (0xFUL << TPI_DEVTYPE_SubType_Pos)
- #define TPI_DEVTYPE_MajorType_Pos 4
- #define TPI_DEVTYPE_MajorType_Msk (0xFUL << TPI_DEVTYPE_MajorType_Pos)

## 6.95.2 Macro Definition Documentation

### 6.95.2.1 #define TPI_ACPR_PRESCALER_Msk (0x1FFFUL << TPI_ACPR_PRESCALER_Pos)

TPI ACPR: PRESCALER Mask

Definition at line 949 of file core_cm4.h.

### 6.95.2.2 #define TPI_ACPR_PRESCALER_Pos 0

TPI ACPR: PRESCALER Position

Definition at line 948 of file core_cm4.h.

### 6.95.2.3 #define TPI_DEVID_AsynClkIn_Msk (0x1UL << TPI_DEVID_AsynClkIn_Pos)

TPI DEVID: AsynClkIn Mask

Definition at line 1049 of file core_cm4.h.

### 6.95.2.4 #define TPI_DEVID_AsynClkIn_Pos 5

TPI DEVID: AsynClkIn Position

Definition at line 1048 of file core_cm4.h.

#### 6.95.2.5 #define TPI_DEVID_MANCVALID_Msk (0x1UL << TPI_DEVID_MANCVALID_Pos)

TPI DEVID: MANCVALID Mask

Definition at line 1040 of file core_cm4.h.

#### 6.95.2.6 #define TPI_DEVID_MANCVALID_Pos 10

TPI DEVID: MANCVALID Position

Definition at line 1039 of file core_cm4.h.

#### 6.95.2.7 #define TPI_DEVID_MinBufSz_Msk (0x7UL << TPI_DEVID_MinBufSz_Pos)

TPI DEVID: MinBufSz Mask

Definition at line 1046 of file core_cm4.h.

#### 6.95.2.8 #define TPI_DEVID_MinBufSz_Pos 6

TPI DEVID: MinBufSz Position

Definition at line 1045 of file core_cm4.h.

#### 6.95.2.9 #define TPI_DEVID_NrTraceInput_Msk (0x1FUL << TPI_DEVID_NrTraceInput_Pos)

TPI DEVID: NrTraceInput Mask

Definition at line 1052 of file core_cm4.h.

#### 6.95.2.10 #define TPI_DEVID_NrTraceInput_Pos 0

TPI DEVID: NrTraceInput Position

Definition at line 1051 of file core_cm4.h.

#### 6.95.2.11 #define TPI_DEVID_NRZVALID_Msk (0x1UL << TPI_DEVID_NRZVALID_Pos)

TPI DEVID: NRZVALID Mask

Definition at line 1037 of file core_cm4.h.

#### 6.95.2.12 #define TPI_DEVID_NRZVALID_Pos 11

TPI DEVID: NRZVALID Position

Definition at line 1036 of file core_cm4.h.

#### 6.95.2.13 #define TPI_DEVID_PTINVALID_Msk (0x1UL << TPI_DEVID_PTINVALID_Pos)

TPI DEVID: PTINVALID Mask

Definition at line 1043 of file core_cm4.h.

#### 6.95.2.14 #define TPI_DEVID_PTINVALID_Pos 9

TPI DEVID: PTINVALID Position

Definition at line 1042 of file core_cm4.h.

#### 6.95.2.15 #define TPI_DEVTYPE_MajorType_Msk (0xFUL << TPI_DEVTYPE_MajorType_Pos)

TPI DEVTYPE: MajorType Mask

Definition at line 1059 of file core_cm4.h.

#### 6.95.2.16 #define TPI_DEVTYPE_MajorType_Pos 4

TPI DEVTYPE: MajorType Position

Definition at line 1058 of file core_cm4.h.

#### 6.95.2.17 #define TPI_DEVTYPE_SubType_Msk (0xFUL << TPI_DEVTYPE_SubType_Pos)

TPI DEVTYPE: SubType Mask

Definition at line 1056 of file core_cm4.h.

#### 6.95.2.18 #define TPI_DEVTYPE_SubType_Pos 0

TPI DEVTYPE: SubType Position

Definition at line 1055 of file core_cm4.h.

#### 6.95.2.19 #define TPI_FFCR_EnFCont_Msk (0x1UL << TPI_FFCR_EnFCont_Pos)

TPI FFCR: EnFCont Mask

Definition at line 973 of file core_cm4.h.

#### 6.95.2.20 #define TPI_FFCR_EnFCont_Pos 1

TPI FFCR: EnFCont Position

Definition at line 972 of file core_cm4.h.

#### 6.95.2.21 #define TPI_FFCR_TrigIn_Msk (0x1UL << TPI_FFCR_TrigIn_Pos)

TPI FFCR: TrigIn Mask

Definition at line 970 of file core_cm4.h.

#### 6.95.2.22 #define TPI_FFCR_TrigIn_Pos 8

TPI FFCR: TrigIn Position

Definition at line 969 of file core_cm4.h.

#### 6.95.2.23 #define TPI_FFSR_FlInProg_Msk (0x1UL << TPI_FFSR_FlInProg_Pos)

TPI FFSR: FlInProg Mask

Definition at line 966 of file core_cm4.h.

#### 6.95.2.24 #define TPI_FFSR_FlInProg_Pos 0

TPI FFSR: FlInProg Position

Definition at line 965 of file core_cm4.h.

#### 6.95.2.25 #define TPI_FFSR_FtNonStop_Msk (0x1UL << TPI_FFSR_FtNonStop_Pos)

TPI FFSR: FtNonStop Mask

Definition at line 957 of file core_cm4.h.

#### 6.95.2.26 #define TPI_FFSR_FtNonStop_Pos 3

TPI FFSR: FtNonStop Position

Definition at line 956 of file core_cm4.h.

#### 6.95.2.27 #define TPI_FFSR_FtStopped_Msk (0x1UL << TPI_FFSR_FtStopped_Pos)

TPI FFSR: FtStopped Mask

Definition at line 963 of file core_cm4.h.

#### 6.95.2.28 #define TPI_FFSR_FtStopped_Pos 1

TPI FFSR: FtStopped Position

Definition at line 962 of file core_cm4.h.

#### 6.95.2.29 #define TPI_FFSR_TCPresent_Msk (0x1UL << TPI_FFSR_TCPresent_Pos)

TPI FFSR: TCPresent Mask

Definition at line 960 of file core_cm4.h.

#### 6.95.2.30 #define TPI_FFSR_TCPresent_Pos 2

TPI FFSR: TCPresent Position

Definition at line 959 of file core_cm4.h.

#### 6.95.2.31 #define TPI_FIFO0_ETM0_Msk (0xFFUL << TPI_FIFO0_ETM0_Pos)

TPI FIFO0: ETM0 Mask

Definition at line 999 of file core_cm4.h.

#### 6.95.2.32 #define TPI_FIFO0_ETM0_Pos 0

TPI FIFO0: ETM0 Position

Definition at line 998 of file core_cm4.h.

#### 6.95.2.33 #define TPI_FIFO0_ETM1_Msk (0xFFUL << TPI_FIFO0_ETM1_Pos)

TPI FIFO0: ETM1 Mask

Definition at line 996 of file core_cm4.h.

#### 6.95.2.34 #define TPI_FIFO0_ETM1_Pos 8

TPI FIFO0: ETM1 Position

Definition at line 995 of file core_cm4.h.

#### 6.95.2.35 #define TPI_FIFO0_ETM2_Msk (0xFFUL << TPI_FIFO0_ETM2_Pos)

TPI FIFO0: ETM2 Mask

Definition at line 993 of file core_cm4.h.

#### 6.95.2.36 #define TPI_FIFO0_ETM2_Pos 16

TPI FIFO0: ETM2 Position

Definition at line 992 of file core_cm4.h.

#### 6.95.2.37 #define TPI_FIFO0_ETM_ATVALID_Msk (0x3UL << TPI_FIFO0_ETM_ATVALID_Pos)

TPI FIFO0: ETM_ATVALID Mask

Definition at line 987 of file core_cm4.h.

#### 6.95.2.38 #define TPI_FIFO0_ETM_ATVALID_Pos 26

TPI FIFO0: ETM_ATVALID Position

Definition at line 986 of file core_cm4.h.

#### 6.95.2.39 #define TPI_FIFO0_ETM_bytecount_Msk (0x3UL << TPI_FIFO0_ETM_bytecount_Pos)

TPI FIFO0: ETM_bytecount Mask

Definition at line 990 of file core_cm4.h.

#### 6.95.2.40 #define TPI_FIFO0_ETM_bytecount_Pos 24

TPI FIFO0: ETM_bytecount Position

Definition at line 989 of file core_cm4.h.

#### 6.95.2.41 #define TPI_FIFO0_ITM_ATVALID_Msk (0x3UL << TPI_FIFO0_ITM_ATVALID_Pos)

TPI FIFO0: ITM_ATVALID Mask

Definition at line 981 of file core_cm4.h.

#### 6.95.2.42 #define TPI_FIFO0_ITM_ATVALID_Pos 29

TPI FIFO0: ITM_ATVALID Position

Definition at line 980 of file core_cm4.h.

#### 6.95.2.43 #define TPI_FIFO0_ITM_bytecount_Msk (0x3UL << TPI_FIFO0_ITM_bytecount_Pos)

TPI FIFO0: ITM_bytecount Mask

Definition at line 984 of file core_cm4.h.

#### 6.95.2.44 #define TPI_FIFO0_ITM_bytecount_Pos 27

TPI FIFO0: ITM_bytecount Position

Definition at line 983 of file core_cm4.h.

#### 6.95.2.45 #define TPI_FIFO1_ETM_ATVALID_Msk (0x3UL << TPI_FIFO1_ETM_ATVALID_Pos)

TPI FIFO1: ETM_ATVALID Mask

Definition at line 1013 of file core_cm4.h.

#### 6.95.2.46 #define TPI_FIFO1_ETM_ATVALID_Pos 26

TPI FIFO1: ETM_ATVALID Position

Definition at line 1012 of file core_cm4.h.

#### 6.95.2.47 #define TPI_FIFO1_ETM_bytecount_Msk (0x3UL << TPI_FIFO1_ETM_bytecount_Pos)

TPI FIFO1: ETM_bytecount Mask

Definition at line 1016 of file core_cm4.h.

#### 6.95.2.48 #define TPI_FIFO1_ETM_bytecount_Pos 24

TPI FIFO1: ETM_bytecount Position

Definition at line 1015 of file core_cm4.h.

#### 6.95.2.49 #define TPI_FIFO1_ITM0_Msk (0xFFUL << TPI_FIFO1_ITM0_Pos)

TPI FIFO1: ITM0 Mask

Definition at line 1025 of file core_cm4.h.

#### 6.95.2.50 #define TPI_FIFO1_ITM0_Pos 0

TPI FIFO1: ITM0 Position

Definition at line 1024 of file core_cm4.h.

#### 6.95.2.51 #define TPI_FIFO1_ITM1_Msk (0xFFUL << TPI_FIFO1_ITM1_Pos)

TPI FIFO1: ITM1 Mask

Definition at line 1022 of file core_cm4.h.

#### 6.95.2.52 #define TPI_FIFO1_ITM1_Pos 8

TPI FIFO1: ITM1 Position

Definition at line 1021 of file core_cm4.h.

#### 6.95.2.53 #define TPI_FIFO1_ITM2_Msk (0xFFUL << TPI_FIFO1_ITM2_Pos)

TPI FIFO1: ITM2 Mask

Definition at line 1019 of file core_cm4.h.

#### 6.95.2.54 #define TPI_FIFO1_ITM2_Pos 16

TPI FIFO1: ITM2 Position

Definition at line 1018 of file core_cm4.h.

#### 6.95.2.55 #define TPI_FIFO1_ITM_ATVALID_Msk (0x3UL << TPI_FIFO1_ITM_ATVALID_Pos)

TPI FIFO1: ITM_ATVALID Mask

Definition at line 1007 of file core_cm4.h.

#### 6.95.2.56 #define TPI_FIFO1_ITM_ATVALID_Pos 29

TPI FIFO1: ITM_ATVALID Position

Definition at line 1006 of file core_cm4.h.

#### 6.95.2.57 #define TPI_FIFO1_ITM_bytecount_Msk (0x3UL << TPI_FIFO1_ITM_bytecount_Pos)

TPI FIFO1: ITM_bytecount Mask

Definition at line 1010 of file core_cm4.h.

#### 6.95.2.58 #define TPI_FIFO1_ITM_bytecount_Pos 27

TPI FIFO1: ITM_bytecount Position

Definition at line 1009 of file core_cm4.h.

#### 6.95.2.59 #define TPI_ITATBCTR0_ATREADY_Msk (0x1UL << TPI_ITATBCTR0_ATREADY_Pos)

TPI ITATBCTR0: ATREADY Mask

Definition at line 1029 of file core_cm4.h.

#### 6.95.2.60 #define TPI_ITATBCTR0_ATREADY_Pos 0

TPI ITATBCTR0: ATREADY Position

Definition at line 1028 of file core_cm4.h.

#### 6.95.2.61 #define TPI_ITATBCTR2_ATREADY_Msk (0x1UL << TPI_ITATBCTR2_ATREADY_Pos)

TPI ITATBCTR2: ATREADY Mask

Definition at line 1003 of file core_cm4.h.

#### 6.95.2.62 #define TPI_ITATBCTR2_ATREADY_Pos 0

TPI ITATBCTR2: ATREADY Position

Definition at line 1002 of file core_cm4.h.

#### 6.95.2.63 #define TPI_ITCTRL_Mode_Msk (0x1UL << TPI_ITCTRL_Mode_Pos)

TPI ITCTRL: Mode Mask

Definition at line 1033 of file core_cm4.h.

#### 6.95.2.64 #define TPI_ITCTRL_Mode_Pos 0

TPI ITCTRL: Mode Position

Definition at line 1032 of file core_cm4.h.

#### 6.95.2.65 #define TPI_SPPR_TXMODE_Msk (0x3UL << TPI_SPPR_TXMODE_Pos)

TPI SPPR: TXMODE Mask

Definition at line 953 of file core_cm4.h.

#### 6.95.2.66 #define TPI_SPPR_TXMODE_Pos 0

TPI SPPR: TXMODE Position

Definition at line 952 of file core_cm4.h.

#### 6.95.2.67 #define TPI_TRIGGER_TRIGGER_Msk (0x1UL << TPI_TRIGGER_TRIGGER_Pos)

TPI TRIGGER: TRIGGER Mask

Definition at line 977 of file core_cm4.h.

#### 6.95.2.68 #define TPI_TRIGGER_TRIGGER_Pos 0

TPI TRIGGER: TRIGGER Position

Definition at line 976 of file core_cm4.h.

## 6.96 USBD_Core

### 6.96.1 Detailed Description

### Data Structures

- struct WB_T
- union WORD_BYTE
- struct BM_T
- union REQUEST_TYPE
- struct USB_SETUP_PACKET
- struct USB_DEVICE_DESCRIPTOR
- struct USB_DEVICE_QUALIFIER_DESCRIPTOR
- struct USB_CONFIGURATION_DESCRIPTOR
- struct USB_IAD_DESCRIPTOR
- struct USB_INTERFACE_DESCRIPTOR
- struct USB_ENDPOINT_DESCRIPTOR
- struct USB_STRING_DESCRIPTOR
- struct USB_COMMON_DESCRIPTOR
- struct USB_OTHER_SPEED_CONFIGURATION

### Macros

- #define USB_CONFIG_POWER_MA(mA) ((mA)/2)
- #define USB_ENDPOINT_0_HS_MAXP 64
- #define USB_ENDPOINT_0_LS_MAXP 8
- #define USB_ENDPOINT_BULK_HS_MAXP 512
- #define WBVAL(x) ((x) & 0xFF),(((x) >> 8) & 0xFF)
- #define B3VAL(x) ((x) & 0xFF),(((x) >> 8) & 0xFF),(((x) >> 16) & 0xFF)
- #define USB_DEVICE_DESC_SIZE (sizeof(USB_DEVICE_DESCRIPTOR))
- #define USB_CONFIGURATION_DESC_SIZE (sizeof(USB_CONFIGURATION_DESCRIPTOR))
- #define USB_INTERFACE_DESC_SIZE (sizeof(USB_INTERFACE_DESCRIPTOR))
- #define USB_INTERFACE_ASSOC_DESC_SIZE (sizeof(USB_IAD_DESCRIPTOR))
- #define USB_ENDPOINT_DESC_SIZE (sizeof(USB_ENDPOINT_DESCRIPTOR))
- #define USB_DEVICE_QUALI_SIZE (sizeof(USB_DEVICE_QUALIFIER_DESCRIPTOR))
- #define USB_OTHER_SPEED_CONF_SIZE (sizeof(USB_OTHER_SPEED_CONFIGURATION))

### Typedefs

- typedef void ∗ USBD_HANDLE_T

### 6.96.2 Macro Definition Documentation

#### 6.96.2.1 #define B3VAL( *x* ) ((x) & 0xFF),(((x) >> 8) & 0xFF),(((x) >> 16) & 0xFF)

Definition at line 693 of file usbd.h.

#### 6.96.2.2 #define REQUEST_CLASS 1

Class Request

Definition at line 110 of file usbd.h.

#### 6.96.2.3 #define REQUEST_DEVICE_TO_HOST 1

Request from device to host

Definition at line 101 of file usbd.h.

#### 6.96.2.4 #define REQUEST_HOST_TO_DEVICE 0

bmRequestType.Dir defines Request from host to device

Definition at line 99 of file usbd.h.

#### 6.96.2.5 #define REQUEST_RESERVED 3

Reserved Request

Definition at line 114 of file usbd.h.

#### 6.96.2.6 #define REQUEST_STANDARD 0

bmRequestType.Type defines Standard Request

Definition at line 108 of file usbd.h.

#### 6.96.2.7 #define REQUEST_TO_DEVICE 0

bmRequestType.Recipient defines Request to device

Definition at line 121 of file usbd.h.

#### 6.96.2.8 #define REQUEST_TO_ENDPOINT 2

Request to endpoint

Definition at line 125 of file usbd.h.

#### 6.96.2.9 #define REQUEST_TO_INTERFACE 1

Request to interface

Definition at line 123 of file usbd.h.

#### 6.96.2.10 #define REQUEST_TO_OTHER 3

Request to other

Definition at line 127 of file usbd.h.

#### 6.96.2.11 #define REQUEST_VENDOR 2

Vendor Request

Definition at line 112 of file usbd.h.

#### 6.96.2.12 #define USB_CONFIG_BUS_POWERED 0x80

Bus powered

Definition at line 288 of file usbd.h.

#### 6.96.2.13 #define USB_CONFIG_POWER_MA( *mA* ) ((mA)/2)

bMaxPower in Configuration Descriptor

Definition at line 296 of file usbd.h.

#### 6.96.2.14 #define USB_CONFIG_POWERED_MASK 0x40

bmAttributes in Configuration Descriptor Power field mask

Definition at line 286 of file usbd.h.

#### 6.96.2.15 #define USB_CONFIG_REMOTE_WAKEUP 0x20

remote wakeup

Definition at line 292 of file usbd.h.

#### 6.96.2.16 #define USB_CONFIG_SELF_POWERED 0xC0

Self powered

Definition at line 290 of file usbd.h.

#### 6.96.2.17 #define USB_CONFIGURATION_DESC_SIZE (sizeof(USB_CONFIGURATION_DESCRIPTOR))

Definition at line 696 of file usbd.h.

#### 6.96.2.18 #define USB_CONFIGURATION_DESCRIPTOR_TYPE 2

Configuration descriptor type

Definition at line 230 of file usbd.h.

#### 6.96.2.19 #define USB_DEBUG_DESCRIPTOR_TYPE 10

Debug descriptor type

Definition at line 246 of file usbd.h.

#### 6.96.2.20 #define USB_DEVICE_CLASS_APP 0xFE

Application device class

Definition at line 277 of file usbd.h.

#### 6.96.2.21 #define USB_DEVICE_CLASS_AUDIO 0x01

Audio device class

Definition at line 257 of file usbd.h.

#### 6.96.2.22 #define USB_DEVICE_CLASS_COMMUNICATIONS 0x02

Communications device class

Definition at line 259 of file usbd.h.

#### 6.96.2.23 #define USB_DEVICE_CLASS_HUB 0x09

Hub device class

Definition at line 273 of file usbd.h.

#### 6.96.2.24 #define USB_DEVICE_CLASS_HUMAN_INTERFACE 0x03

Human interface device class

Definition at line 261 of file usbd.h.

#### 6.96.2.25 #define USB_DEVICE_CLASS_MISCELLANEOUS 0xEF

miscellaneous device class

Definition at line 275 of file usbd.h.

#### 6.96.2.26 #define USB_DEVICE_CLASS_MONITOR 0x04

monitor device class

Definition at line 263 of file usbd.h.

#### 6.96.2.27 #define USB_DEVICE_CLASS_PHYSICAL_INTERFACE 0x05

physical interface device class

Definition at line 265 of file usbd.h.

#### 6.96.2.28 #define USB_DEVICE_CLASS_POWER 0x06

power device class

Definition at line 267 of file usbd.h.

#### 6.96.2.29 #define USB_DEVICE_CLASS_PRINTER 0x07

Printer device class

Definition at line 269 of file usbd.h.

#### 6.96.2.30 #define USB_DEVICE_CLASS_RESERVED 0x00

USB Device Classes Reserved device class

Definition at line 255 of file usbd.h.

#### 6.96.2.31 #define USB_DEVICE_CLASS_STORAGE 0x08

Storage device class

Definition at line 271 of file usbd.h.

#### 6.96.2.32 #define USB_DEVICE_CLASS_VENDOR_SPECIFIC 0xFF

Vendor specific device class

Definition at line 279 of file usbd.h.

#### 6.96.2.33 #define USB_DEVICE_DESC_SIZE (sizeof(USB_DEVICE_DESCRIPTOR))

Definition at line 695 of file usbd.h.

#### 6.96.2.34 #define USB_DEVICE_DESCRIPTOR_TYPE 1

USB Descriptor Types Device descriptor type

Definition at line 228 of file usbd.h.

#### 6.96.2.35 #define USB_DEVICE_QUALI_SIZE (sizeof(USB_DEVICE_QUALIFIER_DESCRIPTOR))

Definition at line 700 of file usbd.h.

#### 6.96.2.36 #define USB_DEVICE_QUALIFIER_DESCRIPTOR_TYPE 6

Device qualifier descriptor type

Definition at line 238 of file usbd.h.

#### 6.96.2.37 #define USB_ENDPOINT_0_HS_MAXP 64

Control endopint EP0's maximum packet size in high-speed mode.

Definition at line 345 of file usbd.h.

#### 6.96.2.38 #define USB_ENDPOINT_0_LS_MAXP 8

Control endopint EP0's maximum packet size in low-speed mode.

Definition at line 347 of file usbd.h.

#### 6.96.2.39 #define USB_ENDPOINT_BULK_HS_MAXP 512

Bulk endopint's maximum packet size in high-speed mode.

Definition at line 349 of file usbd.h.

#### 6.96.2.40 #define USB_ENDPOINT_DESC_SIZE (sizeof(USB_ENDPOINT_DESCRIPTOR))

Definition at line 699 of file usbd.h.

#### 6.96.2.41 #define USB_ENDPOINT_DESCRIPTOR_TYPE 5

Endpoint descriptor type

Definition at line 236 of file usbd.h.

#### 6.96.2.42 #define USB_ENDPOINT_DIRECTION_MASK 0x80

bEndpointAddress in Endpoint Descriptor Endopint address mask

Definition at line 302 of file usbd.h.

#### 6.96.2.43 #define USB_ENDPOINT_IN( *addr* ) ((addr) | 0x80)

Macro to convert IN endopint number to endpoint address value.

Definition at line 306 of file usbd.h.

#### 6.96.2.44 #define USB_ENDPOINT_OUT( *addr* ) ((addr) | 0x00)

Macro to convert OUT endopint number to endpoint address value.

Definition at line 304 of file usbd.h.

#### 6.96.2.45 #define USB_ENDPOINT_SYNC_ADAPTIVE 0x08

Adaptive sync Endopint

Definition at line 329 of file usbd.h.

#### 6.96.2.46 #define USB_ENDPOINT_SYNC_ASYNCHRONOUS 0x04

Asynchronous sync Endopint

Definition at line 327 of file usbd.h.

#### 6.96.2.47 #define USB_ENDPOINT_SYNC_MASK 0x0C

Endopint sync type mask

Definition at line 323 of file usbd.h.

#### 6.96.2.48 #define USB_ENDPOINT_SYNC_NO_SYNCHRONIZATION 0x00

no synchronization Endopint

Definition at line 325 of file usbd.h.

#### 6.96.2.49 #define USB_ENDPOINT_SYNC_SYNCHRONOUS 0x0C

Synchronous sync Endopint

Definition at line 331 of file usbd.h.

#### 6.96.2.50 #define USB_ENDPOINT_TYPE_BULK 0x02

bulk Endopint type

Definition at line 319 of file usbd.h.

#### 6.96.2.51 #define USB_ENDPOINT_TYPE_CONTROL 0x00

Control Endopint type

Definition at line 315 of file usbd.h.

#### 6.96.2.52 #define USB_ENDPOINT_TYPE_INTERRUPT 0x03

interrupt Endopint type

Definition at line 321 of file usbd.h.

#### 6.96.2.53 #define USB_ENDPOINT_TYPE_ISOCHRONOUS 0x01

isochronous Endopint type

Definition at line 317 of file usbd.h.

#### 6.96.2.54 #define USB_ENDPOINT_TYPE_MASK 0x03

bmAttributes in Endpoint Descriptor Endopint type mask

Definition at line 313 of file usbd.h.

#### 6.96.2.55 #define USB_ENDPOINT_USAGE_DATA 0x00

Endopint data usage type

Definition at line 335 of file usbd.h.

#### 6.96.2.56 #define USB_ENDPOINT_USAGE_FEEDBACK 0x10

Endopint feedback usage type

Definition at line 337 of file usbd.h.

#### 6.96.2.57 #define USB_ENDPOINT_USAGE_IMPLICIT_FEEDBACK 0x20

Endopint implicit feedback usage type

Definition at line 339 of file usbd.h.

#### 6.96.2.58 #define USB_ENDPOINT_USAGE_MASK 0x30

Endopint usage type mask

Definition at line 333 of file usbd.h.

#### 6.96.2.59 #define USB_ENDPOINT_USAGE_RESERVED 0x30

Endopint reserved usage type

Definition at line 341 of file usbd.h.

#### 6.96.2.60 #define USB_FEATURE_ENDPOINT_STALL 0

USB Standard Feature selectors ENDPOINT_STALL feature

Definition at line 191 of file usbd.h.

#### 6.96.2.61 #define USB_FEATURE_REMOTE_WAKEUP 1

REMOTE_WAKEUP feature

Definition at line 193 of file usbd.h.

#### 6.96.2.62 #define USB_FEATURE_TEST_MODE 2

TEST_MODE feature

Definition at line 195 of file usbd.h.

#### 6.96.2.63 #define USB_GETSTATUS_ENDPOINT_STALL 0x01

ENDPOINT_STALL status

Definition at line 184 of file usbd.h.

#### 6.96.2.64 #define USB_GETSTATUS_REMOTE_WAKEUP 0x02

REMOTE_WAKEUP capable status

Definition at line 182 of file usbd.h.

#### 6.96.2.65 #define USB_GETSTATUS_SELF_POWERED 0x01

USB GET_STATUS Bit Values SELF_POWERED status

Definition at line 180 of file usbd.h.

#### 6.96.2.66 #define USB_INTERFACE_ASSOC_DESC_SIZE (sizeof(USB_IAD_DESCRIPTOR))

Definition at line 698 of file usbd.h.

#### 6.96.2.67 #define USB_INTERFACE_ASSOCIATION_DESCRIPTOR_TYPE 11

Interface association descriptor type

Definition at line 248 of file usbd.h.

#### 6.96.2.68 #define USB_INTERFACE_DESC_SIZE (sizeof(USB_INTERFACE_DESCRIPTOR))

Definition at line 697 of file usbd.h.

#### 6.96.2.69 #define USB_INTERFACE_DESCRIPTOR_TYPE 4

Interface descriptor type

Definition at line 234 of file usbd.h.

#### 6.96.2.70 #define USB_INTERFACE_POWER_DESCRIPTOR_TYPE 8

Interface power descriptor type

Definition at line 242 of file usbd.h.

#### 6.96.2.71 #define USB_OTG_DESCRIPTOR_TYPE 9

OTG descriptor type

Definition at line 244 of file usbd.h.

#### 6.96.2.72 #define USB_OTHER_SPEED_CONF_SIZE (sizeof(USB_OTHER_SPEED_CONFIGURATION))

Definition at line 701 of file usbd.h.

#### 6.96.2.73 #define USB_OTHER_SPEED_CONFIG_DESCRIPTOR_TYPE 7

Other speed configuration descriptor type

Definition at line 240 of file usbd.h.

#### 6.96.2.74 #define USB_REQUEST_CLEAR_FEATURE 1

CLEAR_FEATURE request

Definition at line 155 of file usbd.h.

#### 6.96.2.75 #define USB_REQUEST_GET_CONFIGURATION 8

GET_CONFIGURATION request

Definition at line 165 of file usbd.h.

#### 6.96.2.76 #define USB_REQUEST_GET_DESCRIPTOR 6

GET_DESCRIPTOR request

Definition at line 161 of file usbd.h.

#### 6.96.2.77 #define USB_REQUEST_GET_INTERFACE 10

GET_INTERFACE request

Definition at line 169 of file usbd.h.

#### 6.96.2.78 #define USB_REQUEST_GET_STATUS 0

USB Standard Request Codes GET_STATUS request

Definition at line 153 of file usbd.h.

#### 6.96.2.79 #define USB_REQUEST_SET_ADDRESS 5

SET_ADDRESS request

Definition at line 159 of file usbd.h.

#### 6.96.2.80 #define USB_REQUEST_SET_CONFIGURATION 9

SET_CONFIGURATION request

Definition at line 167 of file usbd.h.

#### 6.96.2.81 #define USB_REQUEST_SET_DESCRIPTOR 7

SET_DESCRIPTOR request

Definition at line 163 of file usbd.h.

#### 6.96.2.82 #define USB_REQUEST_SET_FEATURE 3

SET_FEATURE request

Definition at line 157 of file usbd.h.

#### 6.96.2.83 #define USB_REQUEST_SET_INTERFACE 11

SET_INTERFACE request

Definition at line 171 of file usbd.h.

#### 6.96.2.84 #define USB_REQUEST_SYNC_FRAME 12

SYNC_FRAME request

Definition at line 173 of file usbd.h.

#### 6.96.2.85 #define USB_STRING_DESCRIPTOR_TYPE 3

String descriptor type

Definition at line 232 of file usbd.h.

#### 6.96.2.86 #define WBVAL( *x* ) ((x) & 0xFF),(((x) >> 8) & 0xFF)

Definition at line 692 of file usbd.h.

### 6.96.3 Typedef Documentation

#### 6.96.3.1 typedef void* USBD_HANDLE_T

USB device stack/module handle.

Definition at line 690 of file usbd.h.

## 6.97 emWin download and installation

The emWin library is not included with LPCOpen. It is available as a separate ZIP file download and installs right into the LPCOpen directory tree for Keil and IAR projects or into the LPCXpresso workspace for LPCXpresso projects. To download emWin, go the the emWin project page at:
`emWin download page on LPCware.com`

Once you have downloaded this files, look at the installation.txt file in the "lpcopen/software/emWin" directory for information on how to install emWin into LPCOPen.

# Chapter 7

# Data Structure Documentation

## 7.1 APSR_Type Union Reference

### 7.1.1 Detailed Description

Union type to access the Application Program Status Register (APSR).

Definition at line 206 of file core_cm0plus.h.

```
#include "core_cm0plus.h"
```

#### Data Fields

- struct {
  uint32_t _reserved0:27
  uint32_t Q:1
  uint32_t V:1
  uint32_t C:1
  uint32_t Z:1
  uint32_t N:1
  } b
- uint32_t w
- struct {
  uint32_t _reserved0:16
  uint32_t GE:4
  uint32_t _reserved1:7
  uint32_t Q:1
  uint32_t V:1
  uint32_t C:1
  uint32_t Z:1
  uint32_t N:1
  } b

### 7.1.2 Field Documentation

#### 7.1.2.1 uint32_t _reserved0

bit: 0..26 Reserved

bit: 0..15 Reserved

Definition at line 211 of file core_cm0plus.h.

#### 7.1.2.2 uint32_t _reserved1

bit: 20..26 Reserved

Definition at line 260 of file core_cm4.h.

#### 7.1.2.3 struct { ... } b

Structure used for bit access

#### 7.1.2.4 struct { ... } b

Structure used for bit access

#### 7.1.2.5 uint32_t C

bit: 29 Carry condition code flag

Definition at line 219 of file core_cm0plus.h.

#### 7.1.2.6 uint32_t GE

bit: 16..19 Greater than or Equal flags

Definition at line 259 of file core_cm4.h.

#### 7.1.2.7 uint32_t N

bit: 31 Negative condition code flag

Definition at line 221 of file core_cm0plus.h.

#### 7.1.2.8 uint32_t Q

bit: 27 Saturation condition flag

Definition at line 217 of file core_cm0plus.h.

#### 7.1.2.9 uint32_t V

bit: 28 Overflow condition code flag

Definition at line 218 of file core_cm0plus.h.

#### 7.1.2.10 uint32_t w

Type used for word access

Definition at line 223 of file core_cm0plus.h.

#### 7.1.2.11 uint32_t Z

bit: 30 Zero condition code flag

Definition at line 220 of file core_cm0plus.h.

The documentation for this union was generated from the following files:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h
- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.2 BM_T Struct Reference

### 7.2.1 Detailed Description

Structure to define 8 bit USB request.

Definition at line 131 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t Recipient: 5
- uint8_t Type: 2
- uint8_t Dir: 1

### 7.2.2 Field Documentation

#### 7.2.2.1 uint8_t Dir

Direction type.

Definition at line 135 of file usbd.h.

#### 7.2.2.2 uint8_t Recipient

Recipient type.

Definition at line 133 of file usbd.h.

#### 7.2.2.3 uint8_t Type

Request type.

Definition at line 134 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.3 CONTROL_Type Union Reference

### 7.3.1 Detailed Description

Union type to access the Control Registers (CONTROL).

Definition at line 268 of file core_cm0plus.h.

```
#include "core_cm0plus.h"
```

## Data Fields

- struct {
  uint32_t nPRIV:1
  uint32_t SPSEL:1
  uint32_t FPCA:1
  uint32_t _reserved0:29
  } b
- uint32_t w
- struct {
  uint32_t nPRIV:1
  uint32_t SPSEL:1
  uint32_t FPCA:1
  uint32_t _reserved0:29
  } b

### 7.3.2 Field Documentation

#### 7.3.2.1 uint32_t _reserved0

bit: 3..31 Reserved

Definition at line 275 of file core_cm0plus.h.

#### 7.3.2.2 struct { ... } b

Structure used for bit access

#### 7.3.2.3 struct { ... } b

Structure used for bit access

#### 7.3.2.4 uint32_t FPCA

bit: 2 FP extension active flag

Definition at line 274 of file core_cm0plus.h.

#### 7.3.2.5 uint32_t nPRIV

bit: 0 Execution privilege in Thread mode

Definition at line 272 of file core_cm0plus.h.

#### 7.3.2.6 uint32_t SPSEL

bit: 1 Stack to be used

Definition at line 273 of file core_cm0plus.h.

#### 7.3.2.7 uint32_t w

Type used for word access

Definition at line 277 of file core_cm0plus.h.

The documentation for this union was generated from the following files:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h
- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.4 CoreDebug_Type Struct Reference

### 7.4.1 Detailed Description

Structure type to access the Core Debug Register (CoreDebug).

Definition at line 1271 of file core_cm4.h.

```
#include "core_cm4.h"
```

### Data Fields

- __IO uint32_t DHCSR
- __O uint32_t DCRSR
- __IO uint32_t DCRDR
- __IO uint32_t DEMCR

### 7.4.2 Field Documentation

#### 7.4.2.1 __IO uint32_t DCRDR

Offset: 0x008 (R/W) Debug Core Register Data Register

Definition at line 1275 of file core_cm4.h.

#### 7.4.2.2 __O uint32_t DCRSR

Offset: 0x004 ( /W) Debug Core Register Selector Register

Definition at line 1274 of file core_cm4.h.

#### 7.4.2.3 __IO uint32_t DEMCR

Offset: 0x00C (R/W) Debug Exception and Monitor Control Register

Definition at line 1276 of file core_cm4.h.

#### 7.4.2.4 __IO uint32_t DHCSR

Offset: 0x000 (R/W) Debug Halting Control and Status Register

Definition at line 1273 of file core_cm4.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.5 DMA_CHDESC_T Struct Reference

### 7.5.1 Detailed Description

Definition at line 523 of file dma_5411x.h.

```
#include "dma_5411x.h"
```

### Data Fields

- uint32_t xfercfg
- uint32_t source
- uint32_t dest
- uint32_t next

### 7.5.2 Field Documentation

#### 7.5.2.1 uint32_t dest

DMA transfer desintation end address

Definition at line 526 of file dma_5411x.h.

#### 7.5.2.2 uint32_t next

Link to next DMA descriptor, must be 16 byte aligned

Definition at line 527 of file dma_5411x.h.

#### 7.5.2.3 uint32_t source

DMA transfer source end address

Definition at line 525 of file dma_5411x.h.

#### 7.5.2.4 uint32_t xfercfg

Transfer configuration (only used in linked lists and ping-pong configs)

Definition at line 524 of file dma_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_5411x.h

## 7.6 DMA_DUAL_DESCRIPTOR_T Struct Reference

### 7.6.1 Detailed Description

Definition at line 55 of file dma_service_5411x.h.

```
#include "dma_service_5411x.h"
```

### Data Fields

- DMA_CHDESC_T descr [2]

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_service_5411x.h

## 7.7 DMA_PERIPHERAL_CONTEXT_T Struct Reference

### 7.7.1 Detailed Description

Definition at line 46 of file dma_service_5411x.h.

```
#include "dma_service_5411x.h"
```

### Data Fields

- uint32_t channel
- volatile uint32_t ∗ register_location
- uint32_t width
- uint32_t src_increment
- uint32_t dst_increment
- bool write

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_service_5411x.h

## 7.8 DMIC_CHANNEL_CONFIG_T Struct Reference

### 7.8.1 Detailed Description

Definition at line 192 of file dmic_5411x.h.

```
#include "dmic_5411x.h"
```

### Data Fields

- STEREO_SIDE_T side
- PDM_DIV_T divhfclk
- uint32_t osr
- int32_t gainshft
- COMPENSATION_T preac2coef

- COMPENSATION_T preac4coef

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dmic_5411x.h

## 7.9 DMIC_STATISTICS_T Struct Reference

### 7.9.1 Detailed Description

DMIC statistics structure.

**Note**

Maintains current DMIC statistics.

Definition at line 87 of file dmic_5411x.h.

```
#include "dmic_5411x.h"
```

### Data Fields

- uint32_t fifo_ints
- uint32_t fifo_overrun
- uint32_t fifo_underrun

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dmic_5411x.h

## 7.10 DWT_Type Struct Reference

### 7.10.1 Detailed Description

Structure type to access the Data Watchpoint and Trace Register (DWT).

Definition at line 774 of file core_cm4.h.

```
#include "core_cm4.h"
```

### Data Fields

- __IO uint32_t CTRL
- __IO uint32_t CYCCNT
- __IO uint32_t CPICNT
- __IO uint32_t EXCCNT
- __IO uint32_t SLEEPCNT
- __IO uint32_t LSUCNT
- __IO uint32_t FOLDCNT
- __I uint32_t PCSR
- __IO uint32_t COMP0
- __IO uint32_t MASK0
- __IO uint32_t FUNCTION0

- uint32_t RESERVED0 [1]
- __IO uint32_t COMP1
- __IO uint32_t MASK1
- __IO uint32_t FUNCTION1
- uint32_t RESERVED1 [1]
- __IO uint32_t COMP2
- __IO uint32_t MASK2
- __IO uint32_t FUNCTION2
- uint32_t RESERVED2 [1]
- __IO uint32_t COMP3
- __IO uint32_t MASK3
- __IO uint32_t FUNCTION3

## 7.10.2 Field Documentation

### 7.10.2.1 __IO uint32_t COMP0

Offset: 0x020 (R/W) Comparator Register 0

Definition at line 784 of file core_cm4.h.

### 7.10.2.2 __IO uint32_t COMP1

Offset: 0x030 (R/W) Comparator Register 1

Definition at line 788 of file core_cm4.h.

### 7.10.2.3 __IO uint32_t COMP2

Offset: 0x040 (R/W) Comparator Register 2

Definition at line 792 of file core_cm4.h.

### 7.10.2.4 __IO uint32_t COMP3

Offset: 0x050 (R/W) Comparator Register 3

Definition at line 796 of file core_cm4.h.

### 7.10.2.5 __IO uint32_t CPICNT

Offset: 0x008 (R/W) CPI Count Register

Definition at line 778 of file core_cm4.h.

### 7.10.2.6 __IO uint32_t CTRL

Offset: 0x000 (R/W) Control Register

Definition at line 776 of file core_cm4.h.

### 7.10.2.7 __IO uint32_t CYCCNT

Offset: 0x004 (R/W) Cycle Count Register

Definition at line 777 of file core_cm4.h.

#### 7.10.2.8 __IO uint32_t EXCCNT

Offset: 0x00C (R/W) Exception Overhead Count Register

Definition at line 779 of file core_cm4.h.

#### 7.10.2.9 __IO uint32_t FOLDCNT

Offset: 0x018 (R/W) Folded-instruction Count Register

Definition at line 782 of file core_cm4.h.

#### 7.10.2.10 __IO uint32_t FUNCTION0

Offset: 0x028 (R/W) Function Register 0

Definition at line 786 of file core_cm4.h.

#### 7.10.2.11 __IO uint32_t FUNCTION1

Offset: 0x038 (R/W) Function Register 1

Definition at line 790 of file core_cm4.h.

#### 7.10.2.12 __IO uint32_t FUNCTION2

Offset: 0x048 (R/W) Function Register 2

Definition at line 794 of file core_cm4.h.

#### 7.10.2.13 __IO uint32_t FUNCTION3

Offset: 0x058 (R/W) Function Register 3

Definition at line 798 of file core_cm4.h.

#### 7.10.2.14 __IO uint32_t LSUCNT

Offset: 0x014 (R/W) LSU Count Register

Definition at line 781 of file core_cm4.h.

#### 7.10.2.15 __IO uint32_t MASK0

Offset: 0x024 (R/W) Mask Register 0

Definition at line 785 of file core_cm4.h.

#### 7.10.2.16 __IO uint32_t MASK1

Offset: 0x034 (R/W) Mask Register 1

Definition at line 789 of file core_cm4.h.

#### 7.10.2.17 __IO uint32_t MASK2

Offset: 0x044 (R/W) Mask Register 2

Definition at line 793 of file core_cm4.h.

#### 7.10.2.18 __IO uint32_t MASK3

Offset: 0x054 (R/W) Mask Register 3

Definition at line 797 of file core_cm4.h.

#### 7.10.2.19 __I uint32_t PCSR

Offset: 0x01C (R/ ) Program Counter Sample Register

Definition at line 783 of file core_cm4.h.

#### 7.10.2.20 uint32_t RESERVED0[1]

Definition at line 787 of file core_cm4.h.

#### 7.10.2.21 uint32_t RESERVED1[1]

Definition at line 791 of file core_cm4.h.

#### 7.10.2.22 uint32_t RESERVED2[1]

Definition at line 795 of file core_cm4.h.

#### 7.10.2.23 __IO uint32_t SLEEPCNT

Offset: 0x010 (R/W) Sleep Count Register

Definition at line 780 of file core_cm4.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.11 I2CM_XFER_T Struct Reference

### 7.11.1 Detailed Description

Master transfer data structure definitions.

Definition at line 74 of file i2cm_5411x.h.

```
#include "i2cm_5411x.h"
```

### Data Fields

- const uint8_t ∗ txBuff
- uint8_t ∗ rxBuff

- uint16_t txSz
- uint16_t rxSz
- uint16_t status
- uint8_t slaveAddr

### 7.11.2 Field Documentation

#### 7.11.2.1 uint8_t∗ rxBuff

Pointer memory where bytes received from I2C be stored

Definition at line 76 of file i2cm_5411x.h.

#### 7.11.2.2 uint16_t rxSz

Number of bytes to received, if 0 only transmission we be carried on

Definition at line 79 of file i2cm_5411x.h.

#### 7.11.2.3 uint8_t slaveAddr

7-bit I2C Slave address

Definition at line 82 of file i2cm_5411x.h.

#### 7.11.2.4 uint16_t status

Status of the current I2C transfer

Definition at line 81 of file i2cm_5411x.h.

#### 7.11.2.5 const uint8_t∗ txBuff

Pointer to array of bytes to be transmitted

Definition at line 75 of file i2cm_5411x.h.

#### 7.11.2.6 uint16_t txSz

Number of bytes in transmit array, if 0 only receive transfer will be carried on

Definition at line 77 of file i2cm_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2cm_5411x.h

## 7.12 I2CS_XFER_T Struct Reference

### 7.12.1 Detailed Description

Slave transfer are performed using 4 callbacks. These 3 callbacks handle most I2C slave transfer cases. When the slave is setup and a slave interrupt is receive and processed with the Chip_I2CS_XferHandler() function in the I2C interrupt handler, one of these 4 callbacks is called. The callbacks can be used for unsized transfers from the master.

When an address is received, the SlaveXferAddr() callback is called with the received address. Only addresses enabled in the slave controller will be handled. The slave controller can support up to 4 slave addresses.

If the master is going to perform a read operation, the SlaveXferSend() callback is called. Place the data byte to send in ∗data and return a value of 0 or I2C_SLVCTL_SLVCONTINUE to the caller, or return a value o I2C_S↩ LVCTL_SLVNACK to NACK the master. If you are using DMA and have setup a DMA descriptor in the callback, return I2C_SLVCTL_SLVDMA.

If the master performs a write operation, the SlaveXferRecv() callback is called with the received data. Return a value of 0 or I2C_SLVCTL_SLVCONTINUE to the caller to continue data transfer. Return I2C_SLVCTL_SLVNACK to NACk the master. If you are using DMA and have setup a DMA descriptor in the callback, return I2C_SLVCTL↩ _SLVDMA.

Once the transfer completes, the SlaveXferDone() callback will be called.

Definition at line 103 of file i2cs_5411x.h.

```
#include "i2cs_5411x.h"
```

## Data Fields

- I2CSlaveXferStart slaveStart
- I2CSlaveXferSend slaveSend
- I2CSlaveXferRecv slaveRecv
- I2CSlaveXferDone slaveDone

## 7.12.2 Field Documentation

### 7.12.2.1 I2CSlaveXferDone slaveDone

Called when a slave transfer is complete

Definition at line 107 of file i2cs_5411x.h.

### 7.12.2.2 I2CSlaveXferRecv slaveRecv

Called when a byte is received from master

Definition at line 106 of file i2cs_5411x.h.

### 7.12.2.3 I2CSlaveXferSend slaveSend

Called when a byte is needed to send to master

Definition at line 105 of file i2cs_5411x.h.

### 7.12.2.4 I2CSlaveXferStart slaveStart

Called when an matching I2C slave address is received

Definition at line 104 of file i2cs_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2cs_5411x.h

## 7.13 I2S_AUDIO_FORMAT_T Struct Reference

### 7.13.1 Detailed Description

Definition at line 182 of file i2s_5411x.h.

```
#include "i2s_5411x.h"
```

### Data Fields

- I2S_DIR_T Direction
- I2S_MSTSLVCFG_T MSCfg
- I2S_MODE_T Mode
- bool RightLow
- bool LeftJust
- bool PDMData
- bool SCKPol
- bool WSPol
- uint32_t Divider
- uint8_t ChannelNumber
- uint16_t WordWidth
- uint16_t FrameWidth
- uint16_t DataPos
- uint16_t FIFOdepth

### 7.13.2 Field Documentation

#### 7.13.2.1 uint8_t ChannelNumber

Channel Number - 1 is mono, 2 is stereo

Definition at line 192 of file i2s_5411x.h.

#### 7.13.2.2 uint16_t DataPos

Data position in the frame

Definition at line 195 of file i2s_5411x.h.

#### 7.13.2.3 I2S_DIR_T Direction

Data direction: tx or rx

Definition at line 183 of file i2s_5411x.h.

#### 7.13.2.4 uint32_t Divider

Flexcomm function clock divider

Definition at line 191 of file i2s_5411x.h.

#### 7.13.2.5 uint16_t FIFOdepth

FIFO depth (fifo config)

Definition at line 196 of file i2s_5411x.h.

#### 7.13.2.6 uint16_t FrameWidth

Frame Width

Definition at line 194 of file i2s_5411x.h.

#### 7.13.2.7 bool LeftJust

left justify data in FIFO

Definition at line 187 of file i2s_5411x.h.

#### 7.13.2.8 I2S_MODE_T Mode

I2S mode

Definition at line 185 of file i2s_5411x.h.

#### 7.13.2.9 I2S_MSTSLVCFG_T MSCfg

Master / Slave configuration

Definition at line 184 of file i2s_5411x.h.

#### 7.13.2.10 bool PDMData

data source is the D-Mic subsystem

Definition at line 188 of file i2s_5411x.h.

#### 7.13.2.11 bool RightLow

right channel data in low portion of FIFO

Definition at line 186 of file i2s_5411x.h.

#### 7.13.2.12 bool SCKPol

SCK polarity

Definition at line 189 of file i2s_5411x.h.

#### 7.13.2.13 uint16_t WordWidth

Word Width

Definition at line 193 of file i2s_5411x.h.

#### 7.13.2.14 bool WSPol

WS polarity

Definition at line 190 of file i2s_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2s_5411x.h

## 7.14 I2S_STATISTICS_T Struct Reference

### 7.14.1 Detailed Description

I2S statistics structure.

**Note**

Maintains current I2S statistics.

Definition at line 204 of file i2s_5411x.h.

```
#include "i2s_5411x.h"
```

### Data Fields

- uint32_t interrupts
- uint32_t lvl_tx
- uint32_t lvl_rx
- uint32_t fifo_err_tx
- uint32_t fifo_err_rx
- uint32_t i2s_busy
- uint32_t i2s_slvfrmerr
- uint32_t i2s_data_paused

### 7.14.2 Field Documentation

#### 7.14.2.1 uint32_t fifo_err_rx

count: FIFO receive errors

Definition at line 209 of file i2s_5411x.h.

#### 7.14.2.2 uint32_t fifo_err_tx

count: FIFO transmit errors

Definition at line 208 of file i2s_5411x.h.

#### 7.14.2.3 uint32_t i2s_busy

count: I2S channel pair is busy

Definition at line 210 of file i2s_5411x.h.

#### 7.14.2.4 uint32_t i2s_data_paused

count: I2S data paused

Definition at line 212 of file i2s_5411x.h.

#### 7.14.2.5 uint32_t i2s_slvfrmerr

count: I2S slave frame error

Definition at line 211 of file i2s_5411x.h.

#### 7.14.2.6 uint32_t interrupts

count: interrupts

Definition at line 205 of file i2s_5411x.h.

#### 7.14.2.7 uint32_t lvl_rx

count: receive interrupts

Definition at line 207 of file i2s_5411x.h.

#### 7.14.2.8 uint32_t lvl_tx

count: transmit interrupts

Definition at line 206 of file i2s_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2s_5411x.h

## 7.15 IPSR_Type Union Reference

### 7.15.1 Detailed Description

Union type to access the Interrupt Program Status Register (IPSR).

Definition at line 229 of file core_cm0plus.h.

```
#include "core_cm0plus.h"
```

### Data Fields

- struct {
    uint32_t ISR:9
    uint32_t _reserved0:23
  } b
- uint32_t w
- struct {
    uint32_t ISR:9
    uint32_t _reserved0:23
  } b

### 7.15.2 Field Documentation

#### 7.15.2.1 uint32_t _reserved0

bit: 9..31 Reserved

Definition at line 234 of file core_cm0plus.h.

#### 7.15.2.2 struct { ... } b

Structure used for bit access

#### 7.15.2.3 struct { ... } b

Structure used for bit access

#### 7.15.2.4 uint32_t ISR

bit: 0.. 8 Exception number

Definition at line 233 of file core_cm0plus.h.

#### 7.15.2.5 uint32_t w

Type used for word access

Definition at line 236 of file core_cm0plus.h.

The documentation for this union was generated from the following files:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h
- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.16 ITM_Type Struct Reference

### 7.16.1 Detailed Description

Structure type to access the Instrumentation Trace Macrocell Register (ITM).

Definition at line 673 of file core_cm4.h.

```
#include "core_cm4.h"
```

### Data Fields

- union {
    __O uint8_t u8
    __O uint16_t u16
    __O uint32_t u32
  } PORT [32]

- uint32_t RESERVED0 [864]
- __IO uint32_t TER
- uint32_t RESERVED1 [15]
- __IO uint32_t TPR
- uint32_t RESERVED2 [15]
- __IO uint32_t TCR
- uint32_t RESERVED3 [29]
- __O uint32_t IWR
- __I uint32_t IRR
- __IO uint32_t IMCR
- uint32_t RESERVED4 [43]
- __O uint32_t LAR

- __I uint32_t LSR
- uint32_t RESERVED5 [6]
- __I uint32_t PID4
- __I uint32_t PID5
- __I uint32_t PID6
- __I uint32_t PID7
- __I uint32_t PID0
- __I uint32_t PID1
- __I uint32_t PID2
- __I uint32_t PID3
- __I uint32_t CID0
- __I uint32_t CID1
- __I uint32_t CID2
- __I uint32_t CID3

## 7.16.2 Field Documentation

### 7.16.2.1 __I uint32_t CID0

Offset: 0xFF0 (R/ ) ITM Component Identification Register #0

Definition at line 703 of file core_cm4.h.

### 7.16.2.2 __I uint32_t CID1

Offset: 0xFF4 (R/ ) ITM Component Identification Register #1

Definition at line 704 of file core_cm4.h.

### 7.16.2.3 __I uint32_t CID2

Offset: 0xFF8 (R/ ) ITM Component Identification Register #2

Definition at line 705 of file core_cm4.h.

### 7.16.2.4 __I uint32_t CID3

Offset: 0xFFC (R/ ) ITM Component Identification Register #3

Definition at line 706 of file core_cm4.h.

### 7.16.2.5 __IO uint32_t IMCR

Offset: 0xF00 (R/W) ITM Integration Mode Control Register

Definition at line 690 of file core_cm4.h.

### 7.16.2.6 __I uint32_t IRR

Offset: 0xEFC (R/ ) ITM Integration Read Register

Definition at line 689 of file core_cm4.h.

#### 7.16.2.7 __O uint32_t IWR

Offset: 0xEF8 ( /W) ITM Integration Write Register

Definition at line 688 of file core_cm4.h.

#### 7.16.2.8 __O uint32_t LAR

Offset: 0xFB0 ( /W) ITM Lock Access Register

Definition at line 692 of file core_cm4.h.

#### 7.16.2.9 __I uint32_t LSR

Offset: 0xFB4 (R/ ) ITM Lock Status Register

Definition at line 693 of file core_cm4.h.

#### 7.16.2.10 __I uint32_t PID0

Offset: 0xFE0 (R/ ) ITM Peripheral Identification Register #0

Definition at line 699 of file core_cm4.h.

#### 7.16.2.11 __I uint32_t PID1

Offset: 0xFE4 (R/ ) ITM Peripheral Identification Register #1

Definition at line 700 of file core_cm4.h.

#### 7.16.2.12 __I uint32_t PID2

Offset: 0xFE8 (R/ ) ITM Peripheral Identification Register #2

Definition at line 701 of file core_cm4.h.

#### 7.16.2.13 __I uint32_t PID3

Offset: 0xFEC (R/ ) ITM Peripheral Identification Register #3

Definition at line 702 of file core_cm4.h.

#### 7.16.2.14 __I uint32_t PID4

Offset: 0xFD0 (R/ ) ITM Peripheral Identification Register #4

Definition at line 695 of file core_cm4.h.

#### 7.16.2.15 __I uint32_t PID5

Offset: 0xFD4 (R/ ) ITM Peripheral Identification Register #5

Definition at line 696 of file core_cm4.h.

#### 7.16.2.16 __I uint32_t PID6

Offset: 0xFD8 (R/ ) ITM Peripheral Identification Register #6

Definition at line 697 of file core_cm4.h.

#### 7.16.2.17 __I uint32_t PID7

Offset: 0xFDC (R/ ) ITM Peripheral Identification Register #7

Definition at line 698 of file core_cm4.h.

#### 7.16.2.18 __O { ... } PORT[32]

Offset: 0x000 ( /W) ITM Stimulus Port Registers

#### 7.16.2.19 uint32_t RESERVED0[864]

Definition at line 681 of file core_cm4.h.

#### 7.16.2.20 uint32_t RESERVED1[15]

Definition at line 683 of file core_cm4.h.

#### 7.16.2.21 uint32_t RESERVED2[15]

Definition at line 685 of file core_cm4.h.

#### 7.16.2.22 uint32_t RESERVED3[29]

Definition at line 687 of file core_cm4.h.

#### 7.16.2.23 uint32_t RESERVED4[43]

Definition at line 691 of file core_cm4.h.

#### 7.16.2.24 uint32_t RESERVED5[6]

Definition at line 694 of file core_cm4.h.

#### 7.16.2.25 __IO uint32_t TCR

Offset: 0xE80 (R/W) ITM Trace Control Register

Definition at line 686 of file core_cm4.h.

#### 7.16.2.26 __IO uint32_t TER

Offset: 0xE00 (R/W) ITM Trace Enable Register

Definition at line 682 of file core_cm4.h.

#### 7.16.2.27 __IO uint32_t TPR

Offset: 0xE40 (R/W) ITM Trace Privilege Register

Definition at line 684 of file core_cm4.h.

#### 7.16.2.28 __O uint16_t u16

Offset: 0x000 ( /W) ITM Stimulus Port 16-bit

Definition at line 678 of file core_cm4.h.

#### 7.16.2.29 __O uint32_t u32

Offset: 0x000 ( /W) ITM Stimulus Port 32-bit

Definition at line 679 of file core_cm4.h.

#### 7.16.2.30 __O uint8_t u8

Offset: 0x000 ( /W) ITM Stimulus Port 8-bit

Definition at line 677 of file core_cm4.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.17 LPC_ADC_T Struct Reference

### 7.17.1 Detailed Description

ADC register block structure.

Definition at line 54 of file adc_5411x.h.

```
#include "adc_5411x.h"
```

### Data Fields

- __IO uint32_t CTRL
- __IO uint32_t INSEL
- union {
      __IO uint32_t SEQ_CTRL [2]
      struct {
          __IO uint32_t SEQA_CTRL
          __IO uint32_t SEQB_CTRL
      }
  };
- union {
      __IO uint32_t SEQ_GDAT [2]
      struct {
          __IO uint32_t SEQA_GDAT
          __IO uint32_t SEQB_GDAT
      }

};

- __IO uint32_t RESERVED0 [2]
- __IO uint32_t DAT [12]
- __IO uint32_t THR_LOW [2]
- __IO uint32_t THR_HIGH [2]
- __IO uint32_t CHAN_THRSEL
- __IO uint32_t INTEN
- __IO uint32_t FLAGS
- __IO uint32_t STARTUP
- __IO uint32_t CALIBR

## 7.17.2 Field Documentation

### 7.17.2.1 union { ... }

### 7.17.2.2 union { ... }

### 7.17.2.3 __IO uint32_t CALIBR

Definition at line 83 of file adc_5411x.h.

### 7.17.2.4 __IO uint32_t CHAN_THRSEL

Definition at line 79 of file adc_5411x.h.

### 7.17.2.5 __IO uint32_t CTRL

< ADCn Structure

Definition at line 55 of file adc_5411x.h.

### 7.17.2.6 __IO uint32_t DAT[12]

Definition at line 76 of file adc_5411x.h.

### 7.17.2.7 __IO uint32_t FLAGS

Definition at line 81 of file adc_5411x.h.

### 7.17.2.8 __IO uint32_t INSEL

Definition at line 56 of file adc_5411x.h.

### 7.17.2.9 __IO uint32_t INTEN

Definition at line 80 of file adc_5411x.h.

### 7.17.2.10 __IO uint32_t RESERVED0[2]

Definition at line 75 of file adc_5411x.h.

#### 7.17.2.11 __IO uint32_t SEQ_CTRL[2]

Definition at line 58 of file adc_5411x.h.

#### 7.17.2.12 __IO uint32_t SEQ_GDAT[2]

Definition at line 67 of file adc_5411x.h.

#### 7.17.2.13 __IO uint32_t SEQA_CTRL

Definition at line 60 of file adc_5411x.h.

#### 7.17.2.14 __IO uint32_t SEQA_GDAT

Definition at line 69 of file adc_5411x.h.

#### 7.17.2.15 __IO uint32_t SEQB_CTRL

Definition at line 61 of file adc_5411x.h.

#### 7.17.2.16 __IO uint32_t SEQB_GDAT

Definition at line 70 of file adc_5411x.h.

#### 7.17.2.17 __IO uint32_t STARTUP

Definition at line 82 of file adc_5411x.h.

#### 7.17.2.18 __IO uint32_t THR_HIGH[2]

Definition at line 78 of file adc_5411x.h.

#### 7.17.2.19 __IO uint32_t THR_LOW[2]

Definition at line 77 of file adc_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/adc_5411x.h

## 7.18 LPC_ASYNC_SYSCON_T Struct Reference

### 7.18.1 Detailed Description

LPC5411X Asynchronous system configuration register block structure.

Definition at line 152 of file syscon_5411x.h.

```
#include "syscon_5411x.h"
```

## Data Fields

- __IO uint32_t AYSNCPRESETCTRL
- __O uint32_t ASYNCPRESETCTRLSET
- __O uint32_t ASYNCPRESETCTRLCLR
- __I uint32_t RESERVED0
- __IO uint32_t ASYNCAPBCLKCTRL
- __IO uint32_t ASYNCAPBCLKCTRLSET
- __IO uint32_t ASYNCAPBCLKCTRLCLR
- __I uint32_t RESERVED1
- __IO uint32_t ASYNCAPBCLKSELA

## 7.18.2 Field Documentation

### 7.18.2.1 __IO uint32_t ASYNCAPBCLKCTRL

clk enable register

Definition at line 157 of file syscon_5411x.h.

### 7.18.2.2 __IO uint32_t ASYNCAPBCLKCTRLCLR

clk enable Clr register

Definition at line 159 of file syscon_5411x.h.

### 7.18.2.3 __IO uint32_t ASYNCAPBCLKCTRLSET

clk enable Set register

Definition at line 158 of file syscon_5411x.h.

### 7.18.2.4 __IO uint32_t ASYNCAPBCLKSELA

clk source mux A register

Definition at line 161 of file syscon_5411x.h.

### 7.18.2.5 __O uint32_t ASYNCPRESETCTRLCLR

peripheral reset Clr register

Definition at line 155 of file syscon_5411x.h.

### 7.18.2.6 __O uint32_t ASYNCPRESETCTRLSET

peripheral reset Set register

Definition at line 154 of file syscon_5411x.h.

### 7.18.2.7 __IO uint32_t AYSNCPRESETCTRL

peripheral reset register

Definition at line 153 of file syscon_5411x.h.

#### 7.18.2.8 __I uint32_t RESERVED0

Definition at line 156 of file syscon_5411x.h.

#### 7.18.2.9 __I uint32_t RESERVED1

Definition at line 160 of file syscon_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/syscon_5411x.h

## 7.19 LPC_CRC_T Struct Reference

### 7.19.1 Detailed Description

CRC register block structure.

Definition at line 47 of file crc_5411x.h.

```
#include "crc_5411x.h"
```

### Data Fields

- __IO uint32_t MODE
- __IO uint32_t SEED
- union {
  __I uint32_t SUM
  __O uint32_t WRDATA32
  __O uint16_t WRDATA16
  __O uint8_t WRDATA8
  };

### 7.19.2 Field Documentation

#### 7.19.2.1 union { ... }

#### 7.19.2.2 __IO uint32_t MODE

< CRC Structure CRC Mode Register

Definition at line 48 of file crc_5411x.h.

#### 7.19.2.3 __IO uint32_t SEED

CRC SEED Register

Definition at line 49 of file crc_5411x.h.

#### 7.19.2.4 __I uint32_t SUM

CRC Checksum Register.

Definition at line 51 of file crc_5411x.h.

#### 7.19.2.5 __O uint16_t WRDATA16

CRC Data Register: write size 16-bit

Definition at line 53 of file crc_5411x.h.

#### 7.19.2.6 __O uint32_t WRDATA32

CRC Data Register: write size 32-bit

Definition at line 52 of file crc_5411x.h.

#### 7.19.2.7 __O uint8_t WRDATA8

CRC Data Register: write size 8-bit

Definition at line 54 of file crc_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/crc_5411x.h

# 7.20 LPC_DMA_CHANNEL_T Struct Reference

## 7.20.1 Detailed Description

DMA Controller shared registers structure.

Definition at line 75 of file dma_5411x.h.

```
#include "dma_5411x.h"
```

### Data Fields

- __IO uint32_t CFG
- __IO uint32_t CTLSTAT
- __IO uint32_t XFERCFG
- __I uint32_t RESERVED

## 7.20.2 Field Documentation

#### 7.20.2.1 __IO uint32_t CFG

< DMA channel register structure DMA Configuration register

Definition at line 76 of file dma_5411x.h.

#### 7.20.2.2 __IO uint32_t CTLSTAT

DMA Control and status register

Definition at line 77 of file dma_5411x.h.

#### 7.20.2.3 __I uint32_t RESERVED

Definition at line 79 of file dma_5411x.h.

#### 7.20.2.4 __IO uint32_t XFERCFG

DMA Transfer configuration register

Definition at line 78 of file dma_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_5411x.h

## 7.21 LPC_DMA_COMMON_T Struct Reference

### 7.21.1 Detailed Description

DMA Controller shared registers structure.

Definition at line 46 of file dma_5411x.h.

```
#include "dma_5411x.h"
```

### Data Fields

- __IO uint32_t ENABLESET
- __I uint32_t RESERVED0
- __O uint32_t ENABLECLR
- __I uint32_t RESERVED1
- __I uint32_t ACTIVE
- __I uint32_t RESERVED2
- __I uint32_t BUSY
- __I uint32_t RESERVED3
- __IO uint32_t ERRINT
- __I uint32_t RESERVED4
- __IO uint32_t INTENSET
- __I uint32_t RESERVED5
- __O uint32_t INTENCLR
- __I uint32_t RESERVED6
- __IO uint32_t INTA
- __I uint32_t RESERVED7
- __IO uint32_t INTB
- __I uint32_t RESERVED8
- __O uint32_t SETVALID
- __I uint32_t RESERVED9
- __O uint32_t SETTRIG
- __I uint32_t RESERVED10
- __O uint32_t ABORT

### 7.21.2 Field Documentation

#### 7.21.2.1 __O uint32_t ABORT

DMA Channel Abort control for all DMA channels

Definition at line 69 of file dma_5411x.h.

#### 7.21.2.2 __I uint32_t ACTIVE

DMA Channel Active status for all DMA channels

Definition at line 51 of file dma_5411x.h.

#### 7.21.2.3 __I uint32_t BUSY

DMA Channel Busy status for all DMA channels

Definition at line 53 of file dma_5411x.h.

#### 7.21.2.4 __O uint32_t ENABLECLR

DMA Channel Enable Clear for all DMA channels

Definition at line 49 of file dma_5411x.h.

#### 7.21.2.5 __IO uint32_t ENABLESET

< DMA shared registers structure DMA Channel Enable read and Set for all DMA channels

Definition at line 47 of file dma_5411x.h.

#### 7.21.2.6 __IO uint32_t ERRINT

DMA Error Interrupt status for all DMA channels

Definition at line 55 of file dma_5411x.h.

#### 7.21.2.7 __IO uint32_t INTA

DMA Interrupt A status for all DMA channels

Definition at line 61 of file dma_5411x.h.

#### 7.21.2.8 __IO uint32_t INTB

DMA Interrupt B status for all DMA channels

Definition at line 63 of file dma_5411x.h.

#### 7.21.2.9 __O uint32_t INTENCLR

DMA Interrupt Enable Clear for all DMA channels

Definition at line 59 of file dma_5411x.h.

#### 7.21.2.10 __IO uint32_t INTENSET

DMA Interrupt Enable read and Set for all DMA channels

Definition at line 57 of file dma_5411x.h.

#### 7.21.2.11 __I uint32_t RESERVED0

Definition at line 48 of file dma_5411x.h.

#### 7.21.2.12 __I uint32_t RESERVED1

Definition at line 50 of file dma_5411x.h.

#### 7.21.2.13 __I uint32_t RESERVED10

Definition at line 68 of file dma_5411x.h.

#### 7.21.2.14 __I uint32_t RESERVED2

Definition at line 52 of file dma_5411x.h.

#### 7.21.2.15 __I uint32_t RESERVED3

Definition at line 54 of file dma_5411x.h.

#### 7.21.2.16 __I uint32_t RESERVED4

Definition at line 56 of file dma_5411x.h.

#### 7.21.2.17 __I uint32_t RESERVED5

Definition at line 58 of file dma_5411x.h.

#### 7.21.2.18 __I uint32_t RESERVED6

Definition at line 60 of file dma_5411x.h.

#### 7.21.2.19 __I uint32_t RESERVED7

Definition at line 62 of file dma_5411x.h.

#### 7.21.2.20 __I uint32_t RESERVED8

Definition at line 64 of file dma_5411x.h.

#### 7.21.2.21 __I uint32_t RESERVED9

Definition at line 66 of file dma_5411x.h.

#### 7.21.2.22 __O uint32_t SETTRIG

DMA Set Trigger control bits for all DMA channels

Definition at line 67 of file dma_5411x.h.

#### 7.21.2.23 __O uint32_t SETVALID

DMA Set ValidPending control bits for all DMA channels

Definition at line 65 of file dma_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_5411x.h

## 7.22 LPC_DMA_T Struct Reference

### 7.22.1 Detailed Description

DMA Controller register block structure.

Definition at line 88 of file dma_5411x.h.

```
#include "dma_5411x.h"
```

### Data Fields

- volatile uint32_t CTRL
- volatile uint32_t INTSTAT
- volatile uint32_t SRAMBASE
- volatile uint32_t RESERVED2 [5]
- LPC_DMA_COMMON_T DMACOMMON [1]
- volatile uint32_t RESERVED0 [225]
- LPC_DMA_CHANNEL_T DMACH [MAX_DMA_CHANNEL]

### 7.22.2 Field Documentation

#### 7.22.2.1 volatile uint32_t CTRL

< DMA Structure DMA control register

Definition at line 89 of file dma_5411x.h.

#### 7.22.2.2 LPC_DMA_CHANNEL_T DMACH[MAX_DMA_CHANNEL]

DMA channel registers

Definition at line 95 of file dma_5411x.h.

#### 7.22.2.3 LPC_DMA_COMMON_T DMACOMMON[1]

DMA shared channel (common) registers

Definition at line 93 of file dma_5411x.h.

#### 7.22.2.4 volatile uint32_t INTSTAT

DMA Interrupt status register

Definition at line 90 of file dma_5411x.h.

#### 7.22.2.5 volatile uint32_t RESERVED0[225]

Definition at line 94 of file dma_5411x.h.

#### 7.22.2.6 volatile uint32_t RESERVED2[5]

Definition at line 92 of file dma_5411x.h.

#### 7.22.2.7 volatile uint32_t SRAMBASE

DMA SRAM address of the channel configuration table

Definition at line 91 of file dma_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_5411x.h

## 7.23 LPC_DMIC_Channel_Type Struct Reference

### 7.23.1 Detailed Description

Definition at line 44 of file dmic_5411x.h.

```
#include "dmic_5411x.h"
```

### Data Fields

- __IO uint32_t OSR
- __IO uint32_t DIVHFCLK
- __IO uint32_t PREAC2FSCOEF
- __IO uint32_t PREAC4FSCOEF
- __IO uint32_t GAINSHFT
- __IO uint32_t TDM96EN
- __IO uint32_t TDM19EN
- __IO uint32_t reserved [25]
- __IO uint32_t FIFO_CTRL
- __IO uint32_t FIFO_STATUS
- __IO uint32_t FIFO_DATA
- __IO uint32_t PHY_CTRL
- __IO uint32_t DC_CTRL
- __IO uint32_t reserved1 [27]

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dmic_5411x.h

## 7.24 LPC_DMIC_T Struct Reference

### 7.24.1 Detailed Description

Definition at line 62 of file dmic_5411x.h.

```
#include "dmic_5411x.h"
```

### Data Fields

- __IO LPC_DMIC_Channel_Type CHANNEL [15]
- __IO uint32_t CHANEN
- __IO uint32_t reserved0 [2]
- __IO uint32_t IOCFG
- __IO uint32_t USE2FS
- __IO uint32_t reserved [27]
- __IO uint32_t HWVADGAIN
- __IO uint32_t HWVADHPFS
- __IO uint32_t HWVADST10
- __IO uint32_t HWVADRSTT
- __IO uint32_t HWVADTHGN
- __IO uint32_t HWVADTHGS
- __IO uint32_t HWVADLOWZ
- __IO uint32_t reserved1 [24]
- __O uint32_t ID

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dmic_5411x.h

## 7.25 LPC_GPIO_T Struct Reference

### 7.25.1 Detailed Description

GPIO port register block structure.

Definition at line 47 of file gpio_5411x.h.

```
#include "gpio_5411x.h"
```

### Data Fields

- __IO uint8_t B [128][32]
- __IO uint32_t W [32][32]
- __IO uint32_t DIR [32]
- __IO uint32_t MASK [32]
- __IO uint32_t PIN [32]
- __IO uint32_t MPIN [32]
- __IO uint32_t SET [32]
- __O uint32_t CLR [32]
- __O uint32_t NOT [32]

### 7.25.2 Field Documentation

#### 7.25.2.1 __IO uint8_t B[128][32]

< GPIO_PORT Structure Offset 0x0000: Byte pin registers ports 0 to n; pins PIOn_0 to PIOn_31

Definition at line 48 of file gpio_5411x.h.

#### 7.25.2.2 __O uint32_t CLR[32]

Offset 0x2280: Clear port n

Definition at line 55 of file gpio_5411x.h.

#### 7.25.2.3 __IO uint32_t DIR[32]

Offset 0x2000: Direction registers port n

Definition at line 50 of file gpio_5411x.h.

#### 7.25.2.4 __IO uint32_t MASK[32]

Offset 0x2080: Mask register port n

Definition at line 51 of file gpio_5411x.h.

#### 7.25.2.5 __IO uint32_t MPIN[32]

Offset 0x2180: Masked port register port n

Definition at line 53 of file gpio_5411x.h.

#### 7.25.2.6 __O uint32_t NOT[32]

Offset 0x2300: Toggle port n

Definition at line 56 of file gpio_5411x.h.

#### 7.25.2.7 __IO uint32_t PIN[32]

Offset 0x2100: Portpin register port n

Definition at line 52 of file gpio_5411x.h.

#### 7.25.2.8 __IO uint32_t SET[32]

Offset 0x2200: Write: Set register for port n Read: output bits for port n

Definition at line 54 of file gpio_5411x.h.

#### 7.25.2.9 __IO uint32_t W[32][32]

Offset 0x1000: Word pin registers port 0 to n

Definition at line 49 of file gpio_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/gpio_5411x.h

## 7.26 LPC_GPIOGROUPINT_T Struct Reference

### 7.26.1 Detailed Description

GPIO grouped interrupt register block structure.

Definition at line 47 of file gpiogroup_5411x.h.

```
#include "gpiogroup_5411x.h"
```

### Data Fields

- __IO uint32_t CTRL
- __I uint32_t RESERVED0 [7]
- __IO uint32_t PORT_POL [8]
- __IO uint32_t PORT_ENA [8]
- uint32_t RESERVED1 [4072]

### 7.26.2 Field Documentation

#### 7.26.2.1 __IO uint32_t CTRL

< GPIO_GROUP_INTn Structure GPIO grouped interrupt control register

Definition at line 48 of file gpiogroup_5411x.h.

#### 7.26.2.2 __IO uint32_t PORT_ENA[8]

GPIO grouped interrupt port m enable register

Definition at line 51 of file gpiogroup_5411x.h.

#### 7.26.2.3 __IO uint32_t PORT_POL[8]

GPIO grouped interrupt port polarity register

Definition at line 50 of file gpiogroup_5411x.h.

#### 7.26.2.4 __I uint32_t RESERVED0[7]

Definition at line 49 of file gpiogroup_5411x.h.

#### 7.26.2.5 uint32_t RESERVED1[4072]

Definition at line 52 of file gpiogroup_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/gpiogroup_5411x.h

## 7.27 LPC_I2S_T Struct Reference

### 7.27.1 Detailed Description

I2S register block structure.

Definition at line 42 of file i2s_5411x.h.

```
#include "i2s_5411x.h"
```

### Data Fields

- __IO uint32_t RESERVED0A [768]
- __IO uint32_t CFG1
- __IO uint32_t CFG2
- __IO uint32_t STAT
- __I uint32_t RESERVED00 [4]
- __IO uint32_t DIV
- __I uint32_t RESERVED0 [120]
- __IO uint32_t FIFOCFG
- __IO uint32_t FIFOSTAT
- __IO uint32_t FIFOTRIG
- __IO uint32_t FIFOINTENSET
- __IO uint32_t FIFOINTENCLR
- __IO uint32_t FIFOINTSTAT
- __O uint32_t FIFOWR
- __I uint32_t FIFORD
- __I uint32_t FIFORDNOPOP
- __I uint32_t FIFORD48HNOPOP
- __I uint32_t RESERVED5 [108]
- __IO uint32_t PSELID
- __I uint32_t PID

## 7.27.2 Field Documentation

### 7.27.2.1 __IO uint32_t CFG1

Offset: 0xC00 Configuration register #1

Definition at line 44 of file i2s_5411x.h.

### 7.27.2.2 __IO uint32_t CFG2

Offset: 0xC04 Configuration register #2

Definition at line 45 of file i2s_5411x.h.

### 7.27.2.3 __IO uint32_t DIV

Offset: 0xC1C Clock divider, used by all channel pairs.

Definition at line 48 of file i2s_5411x.h.

### 7.27.2.4 __IO uint32_t FIFOCFG

I2S FIFO Specific registers Offset: 0xE00 FIFO Configuration register

Definition at line 52 of file i2s_5411x.h.

### 7.27.2.5 __IO uint32_t FIFOINTENCLR

Offset: 0xE14 FIFO Interrupt enable CLEAR register

Definition at line 57 of file i2s_5411x.h.

#### 7.27.2.6 __IO uint32_t FIFOINTENSET

Offset: 0xE10 FIFO Interrupt enable SET register

Definition at line 56 of file i2s_5411x.h.

#### 7.27.2.7 __IO uint32_t FIFOINTSTAT

Offset: 0xE18 FIFO Interrupt Status register

Definition at line 58 of file i2s_5411x.h.

#### 7.27.2.8 __I uint32_t FIFORD

Offset: 0xE30 FIFO Data read register

Definition at line 63 of file i2s_5411x.h.

#### 7.27.2.9 __I uint32_t FIFORD48HNOPOP

Offset: 0xE44 FIFO data read for upper data bits with no FIFO pop

Definition at line 67 of file i2s_5411x.h.

#### 7.27.2.10 __I uint32_t FIFORDNOPOP

Offset: 0xE40 FIFO data read with no FIFO pop

Definition at line 66 of file i2s_5411x.h.

#### 7.27.2.11 __IO uint32_t FIFOSTAT

Offset: 0xE04 FIFO Status register

Definition at line 53 of file i2s_5411x.h.

#### 7.27.2.12 __IO uint32_t FIFOTRIG

Offset: 0xE08 FIFO Trigger level register

Definition at line 54 of file i2s_5411x.h.

#### 7.27.2.13 __O uint32_t FIFOWR

Offset: 0xE20 FIFO Data write register

Definition at line 60 of file i2s_5411x.h.

#### 7.27.2.14 __I uint32_t PID

Offset: 0xFFC Module identification register

Definition at line 72 of file i2s_5411x.h.

#### 7.27.2.15 __IO uint32_t PSELID

FLEXCOMM Interface registers Offset: 0xFF8 Peripheral select/identification register

Definition at line 71 of file i2s_5411x.h.

#### 7.27.2.16 __I uint32_t RESERVED0[120]

Offset: 0xC10 Reserved member

Definition at line 49 of file i2s_5411x.h.

#### 7.27.2.17 __I uint32_t RESERVED00[4]

Offset: 0xC0C Reserved registers

Definition at line 47 of file i2s_5411x.h.

#### 7.27.2.18 __IO uint32_t RESERVED0A[768]

< I2S Structure

Definition at line 43 of file i2s_5411x.h.

#### 7.27.2.19 __I uint32_t RESERVED5[108]

Offset: 0xE48 Reserved register

Definition at line 68 of file i2s_5411x.h.

#### 7.27.2.20 __IO uint32_t STAT

Offset: 0xC08 Status register

Definition at line 46 of file i2s_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2s_5411x.h

## 7.28 LPC_INMUX_T Struct Reference

### 7.28.1 Detailed Description

LPC5411X Input Mux Register Block Structure.

Definition at line 47 of file inmux_5411x.h.

```
#include "inmux_5411x.h"
```

### Data Fields

- __I uint32_t RESERVED1 [48]
- __IO uint32_t PINTSEL [8]
- __IO uint32_t DMA_ITRIG_INMUX [22]
- __I uint32_t RESERVED2 [10]

- __IO uint32_t DMA_OTRIG_INMUX [4]
- __I uint32_t RESERVED3 [4]
- __IO uint32_t FREQMEAS_REF
- __IO uint32_t FREQMEAS_TARGET

## 7.28.2 Field Documentation

### 7.28.2.1 __IO uint32_t DMA_ITRIG_INMUX[22]

Input mux register for DMA trigger inputs

Definition at line 50 of file inmux_5411x.h.

### 7.28.2.2 __IO uint32_t DMA_OTRIG_INMUX[4]

Input mux register for DMA trigger inputs

Definition at line 52 of file inmux_5411x.h.

### 7.28.2.3 __IO uint32_t FREQMEAS_REF

Clock selection for frequency measurement ref clock

Definition at line 54 of file inmux_5411x.h.

### 7.28.2.4 __IO uint32_t FREQMEAS_TARGET

Clock selection for frequency measurement target clock

Definition at line 55 of file inmux_5411x.h.

### 7.28.2.5 __IO uint32_t PINTSEL[8]

Pin interrupt select registers

Definition at line 49 of file inmux_5411x.h.

### 7.28.2.6 __I uint32_t RESERVED1[48]

< INMUX Structure

Definition at line 48 of file inmux_5411x.h.

### 7.28.2.7 __I uint32_t RESERVED2[10]

Definition at line 51 of file inmux_5411x.h.

### 7.28.2.8 __I uint32_t RESERVED3[4]

Definition at line 53 of file inmux_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/inmux_5411x.h

# 7.29 LPC_IOCON_T Struct Reference

## 7.29.1 Detailed Description

LPC5411X IO Configuration Unit register block structure.

Definition at line 47 of file iocon_5411x.h.

```
#include "iocon_5411x.h"
```

### Data Fields

- __IO uint32_t PIO [2][32]

## 7.29.2 Field Documentation

### 7.29.2.1 __IO uint32_t PIO[2][32]

< LPC5411X IOCON Structure

Definition at line 48 of file iocon_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/iocon_5411x.h

# 7.30 LPC_MBOX_T Struct Reference

## 7.30.1 Detailed Description

Mailbox register structure

Definition at line 60 of file mailbox_5411x.h.

```
#include "mailbox_5411x.h"
```

### Data Fields

- LPC_MBOXIRQ_T BOX [MAILBOX_AVAIL]
- LPC_MBOXIRQ_T RESERVED1 [15-MAILBOX_AVAIL]
- __I uint32_t RESERVED2 [2]
- __IO uint32_t MUTEX

## 7.30.2 Field Documentation

### 7.30.2.1 LPC_MBOXIRQ_T BOX[MAILBOX_AVAIL]

< Mailbox register structure Mailbox, offset 0 = M0+, offset 1 = M4

Definition at line 61 of file mailbox_5411x.h.

### 7.30.2.2 __IO uint32_t MUTEX

Mutex

Definition at line 64 of file mailbox_5411x.h.

#### 7.30.2.3 LPC_MBOXIRQ_T RESERVED1[15-MAILBOX_AVAIL]

Definition at line 62 of file mailbox_5411x.h.

#### 7.30.2.4 __I uint32_t RESERVED2[2]

Definition at line 63 of file mailbox_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/mailbox_5411x.h

## 7.31 LPC_MBOXIRQ_T Struct Reference

### 7.31.1 Detailed Description

Individual mailbox IRQ structure

Definition at line 52 of file mailbox_5411x.h.

```
#include "mailbox_5411x.h"
```

### Data Fields

- __IO uint32_t IRQ
- __O uint32_t IRQSET
- __O uint32_t IRQCLR
- __I uint32_t RESERVED

### 7.31.2 Field Documentation

#### 7.31.2.1 __IO uint32_t IRQ

Mailbox data

Definition at line 53 of file mailbox_5411x.h.

#### 7.31.2.2 __O uint32_t IRQCLR

Mailbox dataclearset bits only

Definition at line 55 of file mailbox_5411x.h.

#### 7.31.2.3 __O uint32_t IRQSET

Mailbox data set bits only

Definition at line 54 of file mailbox_5411x.h.

#### 7.31.2.4 __I uint32_t RESERVED

Definition at line 56 of file mailbox_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/mailbox_5411x.h

# 7.32 LPC_MRT_CH_T Struct Reference

## 7.32.1 Detailed Description

MRT register block structure.

Definition at line 53 of file mrt_5411x.h.

```
#include "mrt_5411x.h"
```

## Data Fields

- __IO uint32_t INTVAL
- __O uint32_t TIMER
- __IO uint32_t CTRL
- __IO uint32_t STAT

## 7.32.2 Field Documentation

### 7.32.2.1 __IO uint32_t CTRL

Timer control register

Definition at line 56 of file mrt_5411x.h.

### 7.32.2.2 __IO uint32_t INTVAL

Timer interval register

Definition at line 54 of file mrt_5411x.h.

### 7.32.2.3 __IO uint32_t STAT

Timer status register

Definition at line 57 of file mrt_5411x.h.

### 7.32.2.4 __O uint32_t TIMER

Timer register

Definition at line 55 of file mrt_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/mrt_5411x.h

# 7.33 LPC_MRT_T Struct Reference

## 7.33.1 Detailed Description

MRT register block structure.

Definition at line 63 of file mrt_5411x.h.

```
#include "mrt_5411x.h"
```

## Data Fields

- LPC_MRT_CH_T CHANNEL [MRT_CHANNELS_NUM]
- uint32_t unused [44]
- __IO uint32_t MODCFG
- __O uint32_t IDLE_CH
- __IO uint32_t IRQ_FLAG

### 7.33.2 Field Documentation

#### 7.33.2.1 LPC_MRT_CH_T CHANNEL[MRT_CHANNELS_NUM]

Definition at line 64 of file mrt_5411x.h.

#### 7.33.2.2 __O uint32_t IDLE_CH

Definition at line 67 of file mrt_5411x.h.

#### 7.33.2.3 __IO uint32_t IRQ_FLAG

Definition at line 68 of file mrt_5411x.h.

#### 7.33.2.4 __IO uint32_t MODCFG

Definition at line 66 of file mrt_5411x.h.

#### 7.33.2.5 uint32_t unused[44]

Definition at line 65 of file mrt_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/mrt_5411x.h

## 7.34 LPC_PIN_INT_T Struct Reference

### 7.34.1 Detailed Description

LPC5411X Pin Interrupt and Pattern Match register block structure.

Definition at line 47 of file pinint_5411x.h.

```
#include "pinint_5411x.h"
```

## Data Fields

- __IO uint32_t ISEL
- __IO uint32_t IENR
- __O uint32_t SIENR
- __O uint32_t CIENR
- __IO uint32_t IENF
- __O uint32_t SIENF

- __O uint32_t CIENF
- __IO uint32_t RISE
- __IO uint32_t FALL
- __IO uint32_t IST
- __IO uint32_t PMCTRL
- __IO uint32_t PMSRC
- __IO uint32_t PMCFG

## 7.34.2 Field Documentation

### 7.34.2.1 __O uint32_t CIENF

Clear Pin Interrupt Enable Falling Edge / Active Level address

Definition at line 54 of file pinint_5411x.h.

### 7.34.2.2 __O uint32_t CIENR

Clear Pin Interrupt Enable (Rising) register

Definition at line 51 of file pinint_5411x.h.

### 7.34.2.3 __IO uint32_t FALL

Pin Interrupt Falling Edge register

Definition at line 56 of file pinint_5411x.h.

### 7.34.2.4 __IO uint32_t IENF

Pin Interrupt Enable Falling Edge / Active Level register

Definition at line 52 of file pinint_5411x.h.

### 7.34.2.5 __IO uint32_t IENR

Pin Interrupt Enable (Rising) register

Definition at line 49 of file pinint_5411x.h.

### 7.34.2.6 __IO uint32_t ISEL

< PIN_INT Structure Pin Interrupt Mode register

Definition at line 48 of file pinint_5411x.h.

### 7.34.2.7 __IO uint32_t IST

Pin Interrupt Status register

Definition at line 57 of file pinint_5411x.h.

#### 7.34.2.8 __IO uint32_t PMCFG

GPIO pattern match interrupt bit slice configuration register

Definition at line 60 of file pinint_5411x.h.

#### 7.34.2.9 __IO uint32_t PMCTRL

GPIO pattern match interrupt control register

Definition at line 58 of file pinint_5411x.h.

#### 7.34.2.10 __IO uint32_t PMSRC

GPIO pattern match interrupt bit-slice source register

Definition at line 59 of file pinint_5411x.h.

#### 7.34.2.11 __IO uint32_t RISE

Pin Interrupt Rising Edge register

Definition at line 55 of file pinint_5411x.h.

#### 7.34.2.12 __O uint32_t SIENF

Set Pin Interrupt Enable Falling Edge / Active Level register

Definition at line 53 of file pinint_5411x.h.

#### 7.34.2.13 __O uint32_t SIENR

Set Pin Interrupt Enable (Rising) register

Definition at line 50 of file pinint_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pinint_5411x.h

## 7.35 LPC_PMU_T Struct Reference

### 7.35.1 Detailed Description

PMU register block structure.

**Note**

> Most of the PMU support is handled by the PMU library.

Definition at line 48 of file pmu_5411x.h.

```
#include "pmu_5411x.h"
```

## Data Fields

- __I uint32_t RESERVED0 [17]
- __IO uint32_t BODCTRL

## 7.35.2 Field Documentation

### 7.35.2.1 __IO uint32_t BODCTRL

Definition at line 50 of file pmu_5411x.h.

### 7.35.2.2 __I uint32_t RESERVED0[17]

Definition at line 49 of file pmu_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pmu_5411x.h

# 7.36 LPC_ROM_API_T Struct Reference

## 7.36.1 Detailed Description

High level ROM API structure.

Definition at line 53 of file romapi_5411x.h.

```
#include "romapi_5411x.h"
```

## Data Fields

- const uint32_t usbdApiBase
- const uint32_t reserved_clib
- const uint32_t reserved_power
- const uint32_t reserved_div
- const uint32_t reserved_usart
- const uint32_t reserved_i2cm
- const uint32_t reserved_i2cs
- const uint32_t reserved_i2cmon
- const uint32_t reserved_spim
- const uint32_t reserved_spis
- const uint32_t reserved_dmaaltd
- const uint32_t reserved_adcaltd
- const uint32_t reserved_uartalt
- const uint32_t reserved_flexcomm

## 7.36.2 Field Documentation

### 7.36.2.1 const uint32_t reserved_adcaltd

Reserved

Definition at line 65 of file romapi_5411x.h.

#### 7.36.2.2 const uint32_t reserved_clib

Reserved

Definition at line 55 of file romapi_5411x.h.

#### 7.36.2.3 const uint32_t reserved_div

Reserved

Definition at line 57 of file romapi_5411x.h.

#### 7.36.2.4 const uint32_t reserved_dmaaltd

Reserved

Definition at line 64 of file romapi_5411x.h.

#### 7.36.2.5 const uint32_t reserved_flexcomm

Reserved

Definition at line 67 of file romapi_5411x.h.

#### 7.36.2.6 const uint32_t reserved_i2cm

Reserved

Definition at line 59 of file romapi_5411x.h.

#### 7.36.2.7 const uint32_t reserved_i2cmon

Reserved

Definition at line 61 of file romapi_5411x.h.

#### 7.36.2.8 const uint32_t reserved_i2cs

Reserved

Definition at line 60 of file romapi_5411x.h.

#### 7.36.2.9 const uint32_t reserved_power

Reserved

Definition at line 56 of file romapi_5411x.h.

#### 7.36.2.10 const uint32_t reserved_spim

Reserved

Definition at line 62 of file romapi_5411x.h.

#### 7.36.2.11 const uint32_t reserved_spis

Reserved

Definition at line 63 of file romapi_5411x.h.

#### 7.36.2.12 const uint32_t reserved_uartalt

Reserved

Definition at line 66 of file romapi_5411x.h.

#### 7.36.2.13 const uint32_t reserved_usart

Reserved

Definition at line 58 of file romapi_5411x.h.

#### 7.36.2.14 const uint32_t usbdApiBase

USB API Base

Definition at line 54 of file romapi_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/romapi_5411x.h

## 7.37 LPC_RTC_T Struct Reference

### 7.37.1 Detailed Description

LPC5411X Real Time clock register block structure.

Definition at line 47 of file rtc_5411x.h.

```
#include "rtc_5411x.h"
```

### Data Fields

- __IO uint32_t CTRL
- __IO uint32_t MATCH
- __IO uint32_t COUNT
- __IO uint32_t WAKE

### 7.37.2 Field Documentation

#### 7.37.2.1 __IO uint32_t COUNT

RTC counter register

Definition at line 50 of file rtc_5411x.h.

#### 7.37.2.2 __IO uint32_t CTRL

< RTC RTC control register

Definition at line 48 of file rtc_5411x.h.

#### 7.37.2.3 __IO uint32_t MATCH

PRTC match (alarm) register

Definition at line 49 of file rtc_5411x.h.

#### 7.37.2.4 __IO uint32_t WAKE

RTC high-resolution/wake-up timer control register

Definition at line 51 of file rtc_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/rtc_5411x.h

## 7.38 LPC_SCT_T Struct Reference

### 7.38.1 Detailed Description

State Configurable Timer register block structure.

Definition at line 56 of file sct_5411x.h.

```
#include "sct_5411x.h"
```

### Data Fields

- __IO uint32_t CONFIG
- union {
  __IO uint32_t CTRL_U
  struct {
  __IO uint16_t CTRL_L
  __IO uint16_t CTRL_H
  }
  };
- __IO uint16_t LIMIT_L
- __IO uint16_t LIMIT_H
- __IO uint16_t HALT_L
- __IO uint16_t HALT_H
- __IO uint16_t STOP_L
- __IO uint16_t STOP_H
- __IO uint16_t START_L
- __IO uint16_t START_H
- uint32_t RESERVED1 [10]
- union {
  __IO uint32_t COUNT_U
  struct {
  __IO uint16_t COUNT_L

```
      __IO uint16_t COUNT_H
    }
  };
```

- __IO uint16_t STATE_L
- __IO uint16_t STATE_H
- __I uint32_t INPUT
- __IO uint16_t REGMODE_L
- __IO uint16_t REGMODE_H
- __IO uint32_t OUTPUT
- __IO uint32_t OUTPUTDIRCTRL
- __IO uint32_t RES
- __IO uint32_t DMA0REQUEST
- __IO uint32_t DMA1REQUEST
- uint32_t RESERVED2 [35]
- __IO uint32_t EVEN
- __IO uint32_t EVFLAG
- __IO uint32_t CONEN
- __IO uint32_t CONFLAG
- union {

```
    union {
      uint32_t U
      struct {
        uint16_t L
        uint16_t H
      }
    } MATCH [CONFIG_SCT_nRG]
    union {
      uint32_t U
      struct {
        uint16_t L
        uint16_t H
      }
    } CAP [CONFIG_SCT_nRG]
  };
```

- uint32_t RESERVED3 [48+(16-CONFIG_SCT_nRG)]
- union {

```
    union {
      uint32_t U
      struct {
        uint16_t L
        uint16_t H
      }
    } MATCHREL [CONFIG_SCT_nRG]
    union {
      uint32_t U
      struct {
        uint16_t L
        uint16_t H
      }
    } CAPCTRL [CONFIG_SCT_nRG]
  };
```

- uint32_t RESERVED6 [48+(16-CONFIG_SCT_nRG)]

- struct {
    uint32_t STATE
    uint32_t CTRL
  } EVENT [CONFIG_SCT_nEV]
- uint32_t RESERVED9 [128-2 ∗CONFIG_SCT_nEV]
- struct {
    uint32_t SET
    uint32_t CLR
  } OUT [CONFIG_SCT_nOU]
- uint32_t RESERVED10 [191-2 ∗CONFIG_SCT_nOU]
- __I uint32_t MODULECONTENT

## 7.38.2 Field Documentation

### 7.38.2.1 union { ... }

### 7.38.2.2 union { ... }

### 7.38.2.3 union { ... }

### 7.38.2.4 union { ... }

### 7.38.2.5 __I { ... } CAP[CONFIG_SCT_nRG]

### 7.38.2.6 __IO { ... } CAPCTRL[CONFIG_SCT_nRG]

### 7.38.2.7 uint32_t CLR

Output n Clear Register

Definition at line 166 of file sct_5411x.h.

### 7.38.2.8 __IO uint32_t CONEN

conflict enable register

Definition at line 102 of file sct_5411x.h.

### 7.38.2.9 __IO uint32_t CONFIG

Configuration Register

Definition at line 57 of file sct_5411x.h.

### 7.38.2.10 __IO uint32_t CONFLAG

conflict flag register

Definition at line 103 of file sct_5411x.h.

### 7.38.2.11 __IO uint16_t COUNT_H

counter register for counter H

Definition at line 84 of file sct_5411x.h.

#### 7.38.2.12 __IO uint16_t COUNT_L

counter register for counter L

Definition at line 83 of file sct_5411x.h.

#### 7.38.2.13 __IO uint32_t COUNT_U

counter register

Definition at line 80 of file sct_5411x.h.

#### 7.38.2.14 uint32_t CTRL

Definition at line 159 of file sct_5411x.h.

#### 7.38.2.15 __IO uint16_t CTRL_H

High control register

Definition at line 64 of file sct_5411x.h.

#### 7.38.2.16 __IO uint16_t CTRL_L

Low control register

Definition at line 63 of file sct_5411x.h.

#### 7.38.2.17 __IO uint32_t CTRL_U

Control Register

Definition at line 60 of file sct_5411x.h.

#### 7.38.2.18 __IO uint32_t DMA0REQUEST

DMA0 Request Register

Definition at line 97 of file sct_5411x.h.

#### 7.38.2.19 __IO uint32_t DMA1REQUEST

DMA1 Request Register

Definition at line 98 of file sct_5411x.h.

#### 7.38.2.20 __IO uint32_t EVEN

event enable register

Definition at line 100 of file sct_5411x.h.

#### 7.38.2.21 __IO { ... } EVENT[CONFIG_SCT_nEV]

#### 7.38.2.22 __IO uint32_t EVFLAG

event flag register

Definition at line 101 of file sct_5411x.h.

#### 7.38.2.23 uint16_t H

SCTMATCH[i].H Access to H value

SCTCAP[i].H Access to H value

Definition at line 112 of file sct_5411x.h.

#### 7.38.2.24 __IO uint16_t HALT_H

halt register for counter H

Definition at line 72 of file sct_5411x.h.

#### 7.38.2.25 __IO uint16_t HALT_L

halt register for counter L

Definition at line 71 of file sct_5411x.h.

#### 7.38.2.26 __I uint32_t INPUT

input register

Definition at line 91 of file sct_5411x.h.

#### 7.38.2.27 uint16_t L

SCTMATCH[i].L Access to L value

SCTCAP[i].L Access to L value

Definition at line 111 of file sct_5411x.h.

#### 7.38.2.28 __IO uint16_t LIMIT_H

limit register for counter H

Definition at line 70 of file sct_5411x.h.

#### 7.38.2.29 __IO uint16_t LIMIT_L

limit register for counter L

Definition at line 69 of file sct_5411x.h.

#### 7.38.2.30 __IO { ... } MATCH[CONFIG_SCT_nRG]

#### 7.38.2.31 __IO { ... } MATCHREL[CONFIG_SCT_nRG]

#### 7.38.2.32 __I uint32_t MODULECONTENT

0x7FC Module Content

Definition at line 170 of file sct_5411x.h.

#### 7.38.2.33 __IO { ... } OUT[CONFIG_SCT_nOU]

#### 7.38.2.34 __IO uint32_t OUTPUT

output register

Definition at line 94 of file sct_5411x.h.

#### 7.38.2.35 __IO uint32_t OUTPUTDIRCTRL

output counter direction Control Register

Definition at line 95 of file sct_5411x.h.

#### 7.38.2.36 __IO uint16_t REGMODE_H

match - capture registers mode register H

Definition at line 93 of file sct_5411x.h.

#### 7.38.2.37 __IO uint16_t REGMODE_L

match - capture registers mode register L

Definition at line 92 of file sct_5411x.h.

#### 7.38.2.38 __IO uint32_t RES

conflict resolution register

Definition at line 96 of file sct_5411x.h.

#### 7.38.2.39 uint32_t RESERVED1[10]

0x03C reserved

Definition at line 77 of file sct_5411x.h.

#### 7.38.2.40 uint32_t RESERVED10[191-2 ∗CONFIG_SCT_nOU]

...-0x7F8 reserved

Definition at line 169 of file sct_5411x.h.

#### 7.38.2.41 uint32_t RESERVED2[35]

Definition at line 99 of file sct_5411x.h.

#### 7.38.2.42 uint32_t RESERVED3[48+(16-CONFIG_SCT_nRG)]

Definition at line 129 of file sct_5411x.h.

#### 7.38.2.43 uint32_t RESERVED6[48+(16-CONFIG_SCT_nRG)]

Definition at line 155 of file sct_5411x.h.

#### 7.38.2.44 uint32_t RESERVED9[128-2 ∗CONFIG_SCT_nEV]

...-0x4FC reserved

Definition at line 162 of file sct_5411x.h.

#### 7.38.2.45 uint32_t SET

< 0x500-0x57C SCTOUT[i].SET / SCTOUT[i].CLR Output n Set Register

Definition at line 165 of file sct_5411x.h.

#### 7.38.2.46 __IO uint16_t START_H

start register for counter H

Definition at line 76 of file sct_5411x.h.

#### 7.38.2.47 __IO uint16_t START_L

start register for counter L

Definition at line 75 of file sct_5411x.h.

#### 7.38.2.48 uint32_t STATE

Definition at line 158 of file sct_5411x.h.

#### 7.38.2.49 __IO uint16_t STATE_H

state register for counter H

Definition at line 90 of file sct_5411x.h.

#### 7.38.2.50 __IO uint16_t STATE_L

state register for counter L

Definition at line 89 of file sct_5411x.h.

#### 7.38.2.51 __IO uint16_t STOP_H

stop register for counter H

Definition at line 74 of file sct_5411x.h.

#### 7.38.2.52 __IO uint16_t STOP_L

stop register for counter L

Definition at line 73 of file sct_5411x.h.

#### 7.38.2.53 uint32_t U

< ... Match / Capture value SCTMATCH[i].U Unified 32-bit register

SCTCAP[i].U Unified 32-bit register

Definition at line 108 of file sct_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/sct_5411x.h

## 7.39 LPC_SPI_T Struct Reference

### 7.39.1 Detailed Description

SPI register block structure.

Definition at line 55 of file spi_common_5411x.h.

```
#include "spi_common_5411x.h"
```

### Data Fields

- __I uint32_t RESERVED0 [256]
- __IO uint32_t CFG
- __IO uint32_t DLY
- __IO uint32_t STAT
- __IO uint32_t INTENSET
- __IO uint32_t INTENCLR
- __I uint32_t RESERVED1 [4]
- __IO uint32_t DIV
- __IO uint32_t INTSTAT
- __IO uint32_t FIFOCFG
- __IO uint32_t FIFOSTAT
- __IO uint32_t FIFOTRIG
- __IO uint32_t FIFOINTENSET
- __IO uint32_t FIFOINTENCLR
- __IO uint32_t FIFOINTSTAT
- __O uint32_t FIFOWR
- __I uint32_t FIFORD
- __I uint32_t FIFORDNOPOP
- __IO uint32_t PSELID
- __I uint32_t PID

### 7.39.2 Field Documentation

#### 7.39.2.1 __IO uint32_t CFG

Offset: 0x400 SPI Configuration register

Definition at line 58 of file spi_common_5411x.h.

#### 7.39.2.2 __IO uint32_t DIV

Offset: 0x424 SPI clock Divider register

Definition at line 64 of file spi_common_5411x.h.

#### 7.39.2.3 __IO uint32_t DLY

Offset: 0x404 SPI Delay register

Definition at line 59 of file spi_common_5411x.h.

#### 7.39.2.4 __IO uint32_t FIFOCFG

Offset: 0xE00 FIFO Configuration register

Definition at line 69 of file spi_common_5411x.h.

#### 7.39.2.5 __IO uint32_t FIFOINTENCLR

Offset: 0xE14 FIFO Interrupt enable CLEAR register

Definition at line 74 of file spi_common_5411x.h.

#### 7.39.2.6 __IO uint32_t FIFOINTENSET

Offset: 0xE10 FIFO Interrupt enable SET register

Definition at line 73 of file spi_common_5411x.h.

#### 7.39.2.7 __IO uint32_t FIFOINTSTAT

Offset: 0xE18 FIFO Interrupt Status register

Definition at line 75 of file spi_common_5411x.h.

#### 7.39.2.8 __I uint32_t FIFORD

Offset: 0xE30 FIFO Data read register

Definition at line 79 of file spi_common_5411x.h.

#### 7.39.2.9 __I uint32_t FIFORDNOPOP

Offset: 0xE40 FIFO Data peek (read without popping out of queue) register

Definition at line 81 of file spi_common_5411x.h.

#### 7.39.2.10 __IO uint32_t FIFOSTAT

Offset: 0xE04 FIFO Status register

Definition at line 70 of file spi_common_5411x.h.

#### 7.39.2.11 __IO uint32_t FIFOTRIG

Offset: 0xE08 FIFO Trigger level register

Definition at line 71 of file spi_common_5411x.h.

#### 7.39.2.12 __O uint32_t FIFOWR

Offset: 0xE20 FIFO Data write register

Definition at line 77 of file spi_common_5411x.h.

#### 7.39.2.13 __IO uint32_t INTENCLR

Offset: 0x410 SPI Interrupt Enable Clear register

Definition at line 62 of file spi_common_5411x.h.

#### 7.39.2.14 __IO uint32_t INTENSET

Offset: 0x40c SPI Interrupt Enable Set register

Definition at line 61 of file spi_common_5411x.h.

#### 7.39.2.15 __IO uint32_t INTSTAT

Offset: 0x428 SPI Interrupt Status register

Definition at line 65 of file spi_common_5411x.h.

#### 7.39.2.16 __I uint32_t PID

Offset: 0xFFC Module identification register

Definition at line 86 of file spi_common_5411x.h.

#### 7.39.2.17 __IO uint32_t PSELID

Offset: 0xFF8 Peripheral select/identification register

Definition at line 85 of file spi_common_5411x.h.

#### 7.39.2.18 __I uint32_t RESERVED0[256]

< SPI Structure SPI registers

Definition at line 57 of file spi_common_5411x.h.

#### 7.39.2.19 __I uint32_t RESERVED1[4]

Definition at line 63 of file spi_common_5411x.h.

#### 7.39.2.20 __IO uint32_t STAT

Offset: 0x408 SPI Status register

Definition at line 60 of file spi_common_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spi_common_5411x.h

## 7.40 LPC_SYSCON_T Struct Reference

### 7.40.1 Detailed Description

LPC5411X Main system configuration register block structure.

Definition at line 47 of file syscon_5411x.h.

```
#include "syscon_5411x.h"
```

### Data Fields

- __IO uint32_t SYSMEMREMAP
- __I uint32_t RESERVED0 [3]
- __IO uint32_t AHBMATPRIO
- __I uint32_t RESERVED1 [11]
- __IO uint32_t SYSTCKCAL
- __I uint32_t RESERVED2 [1]
- __IO uint32_t NMISRC
- __IO uint32_t ASYNCAPBCTRL
- __I uint32_t RESERVED3 [28]
- __I uint32_t PIOPORCAP [2]
- __I uint32_t RESERVED4 [2]
- __I uint32_t PIORESCAP [2]
- __I uint32_t RESERVED5 [10]
- __IO uint32_t PRESETCTRL [2]
- __I uint32_t RESERVED6 [6]
- __O uint32_t PRESETCTRLSET [2]
- __I uint32_t RESERVED7 [6]
- __O uint32_t PRESETCTRLCLR [2]
- __I uint32_t RESERVED8 [42]
- __IO uint32_t SYSRSTSTAT
- __I uint32_t RESERVED9 [3]
- __IO uint32_t AHBCLKCTRL [2]
- __I uint32_t RESERVED10 [6]
- __O uint32_t AHBCLKCTRLSET [2]
- __I uint32_t RESERVED11 [6]
- __O uint32_t AHBCLKCTRLCLR [2]
- __I uint32_t RESERVED12 [14]
- __IO uint32_t MAINCLKSELA
- __IO uint32_t MAINCLKSELB
- __IO uint32_t CLKOUTSELA
- __I uint32_t RESERVED13
- __IO uint32_t SYSPLLCLKSEL
- __I uint32_t RESERVED14 [3]

- __IO uint32_t SPIFICLKSEL
- __IO uint32_t ADCCLKSEL
- __IO uint32_t USBCLKSEL
- __I uint32_t RESERVED15
- __IO uint32_t FXCOMCLKSEL [8]
- __I uint32_t RESERVED16 [4]
- __IO uint32_t MCLKCLKSEL
- __I uint32_t RESERVED16A
- __IO uint32_t FRGCLKSEL
- __IO uint32_t DMICCLKSEL
- __I uint32_t RESERVED17 [4]
- __IO uint32_t SYSTICKCLKDIV
- __I uint32_t RESERVED18 [31]
- __IO uint32_t AHBCLKDIV
- __IO uint32_t CLKOUTDIV
- __I uint32_t RESERVED19 [2]
- __IO uint32_t SPIFICLKDIV
- __IO uint32_t ADCCLKDIV
- __IO uint32_t USBCLKDIV
- __I uint32_t RESERVED20
- __IO uint32_t FRGCTRL
- __I uint32_t RESERVED21
- __IO uint32_t DMICCLKDIV
- __IO uint32_t MCLKDIV
- __I uint32_t RESERVED22 [20]
- __IO uint32_t FLASHCFG
- __I uint32_t RESERVED23 [2]
- __IO uint32_t USBCLKCTRL
- __IO uint32_t USBCLKSTAT
- __I uint32_t RESERVED24
- __IO uint32_t FREQMECTRL
- __I uint32_t RESERVED25
- __IO uint32_t MCLKIO
- __I uint32_t RESERVED26 [55]
- __IO uint32_t FROCTRL
- __I uint32_t RESERVED27
- __IO uint32_t WDTOSCCTRL
- __IO uint32_t RTCOSCCTRL
- __I uint32_t RESERVED28 [28]
- __IO uint32_t SYSPLLCTRL
- __I uint32_t SYSPLLSTAT
- __IO uint32_t SYSPLLNDEC
- __IO uint32_t SYSPLLPDEC
- __IO uint32_t SYSPLLSSCTRL [2]
- __I uint32_t RESERVED29 [30]
- __IO uint32_t PDRUNCFG [2]
- __I uint32_t RESERVED30 [2]
- __O uint32_t PDRUNCFGSET [2]
- __I uint32_t RESERVED31 [2]
- __O uint32_t PDRUNCFGCLR [2]
- __I uint32_t RESERVED32 [18]
- __IO uint32_t STARTERP [2]
- __I uint32_t RESERVED33 [6]
- __O uint32_t STARTERPSET [2]
- __I uint32_t RESERVED34 [6]

- __O uint32_t STARTERPCLR [2]
- __I uint32_t RESERVED35 [78]
- __IO uint32_t CPCTRL
- __IO uint32_t CPBOOT
- __IO uint32_t CPSTACK
- __I uint32_t CPSTAT
- __I uint32_t RESERVED36 [381]
- __IO uint32_t AUTOCGOR
- __I uint32_t RESERVED37 [123]
- __I uint32_t JTAGIDCODE
- __I uint32_t DEVICE_ID [2]

## 7.40.2 Field Documentation

### 7.40.2.1 __IO uint32_t ADCCLKDIV

ADC Clock divider register

Definition at line 98 of file syscon_5411x.h.

### 7.40.2.2 __IO uint32_t ADCCLKSEL

ADC Async Clk Sel register

Definition at line 82 of file syscon_5411x.h.

### 7.40.2.3 __IO uint32_t AHBCLKCTRL[2]

AHB Peripheral Clk Enable register

Definition at line 69 of file syscon_5411x.h.

### 7.40.2.4 __O uint32_t AHBCLKCTRLCLR[2]

AHB Peripheral Clk Enable Clr register

Definition at line 73 of file syscon_5411x.h.

### 7.40.2.5 __O uint32_t AHBCLKCTRLSET[2]

AHB Peripheral Clk Enable Set register

Definition at line 71 of file syscon_5411x.h.

### 7.40.2.6 __IO uint32_t AHBCLKDIV

AHB Clock divider

Definition at line 94 of file syscon_5411x.h.

### 7.40.2.7 __IO uint32_t AHBMATPRIO

AHB Martix priority register

Definition at line 50 of file syscon_5411x.h.

#### 7.40.2.8 __IO uint32_t ASYNCAPBCTRL

Asynch APB chiplet control register

Definition at line 55 of file syscon_5411x.h.

#### 7.40.2.9 __IO uint32_t AUTOCGOR

Definition at line 143 of file syscon_5411x.h.

#### 7.40.2.10 __IO uint32_t CLKOUTDIV

CLKOUT divider

Definition at line 95 of file syscon_5411x.h.

#### 7.40.2.11 __IO uint32_t CLKOUTSELA

Clk Out Sel Source B register

Definition at line 77 of file syscon_5411x.h.

#### 7.40.2.12 __IO uint32_t CPBOOT

Coprocessor boot address

Definition at line 139 of file syscon_5411x.h.

#### 7.40.2.13 __IO uint32_t CPCTRL

Coprocessor control register

Definition at line 138 of file syscon_5411x.h.

#### 7.40.2.14 __IO uint32_t CPSTACK

Coprocessor stack address register

Definition at line 140 of file syscon_5411x.h.

#### 7.40.2.15 __I uint32_t CPSTAT

Coprocessor status register

Definition at line 141 of file syscon_5411x.h.

#### 7.40.2.16 __I uint32_t DEVICE_ID[2]

Device ID Registers

Definition at line 146 of file syscon_5411x.h.

#### 7.40.2.17 __IO uint32_t DMICCLKDIV

DMIC Clock divider register

Definition at line 103 of file syscon_5411x.h.

#### 7.40.2.18 __IO uint32_t DMICCLKSEL

DMIC Clock select register

Definition at line 90 of file syscon_5411x.h.

#### 7.40.2.19 __IO uint32_t FLASHCFG

Flash wait state configuration register

Definition at line 106 of file syscon_5411x.h.

#### 7.40.2.20 __IO uint32_t FREQMECTRL

Frequency measure register

Definition at line 111 of file syscon_5411x.h.

#### 7.40.2.21 __IO uint32_t FRGCLKSEL

FRG Clock select register

Definition at line 89 of file syscon_5411x.h.

#### 7.40.2.22 __IO uint32_t FRGCTRL

Fraction Rate Generator Ctrl register

Definition at line 101 of file syscon_5411x.h.

#### 7.40.2.23 __IO uint32_t FROCTRL

FRO oscillator control register

Definition at line 115 of file syscon_5411x.h.

#### 7.40.2.24 __IO uint32_t FXCOMCLKSEL[8]

FlexCOM CLK sel register

Definition at line 85 of file syscon_5411x.h.

#### 7.40.2.25 __I uint32_t JTAGIDCODE

JTAG ID Code register

Definition at line 145 of file syscon_5411x.h.

#### 7.40.2.26 __IO uint32_t MAINCLKSELA

Main Clk sel Source Sel A register

Definition at line 75 of file syscon_5411x.h.

#### 7.40.2.27 __IO uint32_t MAINCLKSELB

Main Clk sel Source Sel B register

Definition at line 76 of file syscon_5411x.h.

#### 7.40.2.28 __IO uint32_t MCLKCLKSEL

MCLK Clock select register

Definition at line 87 of file syscon_5411x.h.

#### 7.40.2.29 __IO uint32_t MCLKDIV

I2S MClock divider register

Definition at line 104 of file syscon_5411x.h.

#### 7.40.2.30 __IO uint32_t MCLKIO

MCLK Input Output register

Definition at line 113 of file syscon_5411x.h.

#### 7.40.2.31 __IO uint32_t NMISRC

NMI Source select register

Definition at line 54 of file syscon_5411x.h.

#### 7.40.2.32 __IO uint32_t PDRUNCFG[2]

Power Down configuration registers

Definition at line 126 of file syscon_5411x.h.

#### 7.40.2.33 __O uint32_t PDRUNCFGCLR[2]

Power down configuartion clear register

Definition at line 130 of file syscon_5411x.h.

#### 7.40.2.34 __O uint32_t PDRUNCFGSET[2]

Power down configuartion set register

Definition at line 128 of file syscon_5411x.h.

#### 7.40.2.35 __I uint32_t PIOPORCAP[2]

Power on Reset; port capture register

Definition at line 57 of file syscon_5411x.h.

#### 7.40.2.36 __I uint32_t PIORESCAP[2]

Reset; port capture register

Definition at line 59 of file syscon_5411x.h.

#### 7.40.2.37 __IO uint32_t PRESETCTRL[2]

Peripheral Reset control

Definition at line 61 of file syscon_5411x.h.

#### 7.40.2.38 __O uint32_t PRESETCTRLCLR[2]

Peripheral Reset Control set

Definition at line 65 of file syscon_5411x.h.

#### 7.40.2.39 __O uint32_t PRESETCTRLSET[2]

Peripheral Reset Control set

Definition at line 63 of file syscon_5411x.h.

#### 7.40.2.40 __I uint32_t RESERVED0[3]

Definition at line 49 of file syscon_5411x.h.

#### 7.40.2.41 __I uint32_t RESERVED1[11]

Definition at line 51 of file syscon_5411x.h.

#### 7.40.2.42 __I uint32_t RESERVED10[6]

Definition at line 70 of file syscon_5411x.h.

#### 7.40.2.43 __I uint32_t RESERVED11[6]

Definition at line 72 of file syscon_5411x.h.

#### 7.40.2.44 __I uint32_t RESERVED12[14]

Definition at line 74 of file syscon_5411x.h.

#### 7.40.2.45 __I uint32_t RESERVED13

Definition at line 78 of file syscon_5411x.h.

#### 7.40.2.46 __I uint32_t RESERVED14[3]

Definition at line 80 of file syscon_5411x.h.

#### 7.40.2.47 __I uint32_t RESERVED15

Definition at line 84 of file syscon_5411x.h.

#### 7.40.2.48 __I uint32_t RESERVED16[4]

Definition at line 86 of file syscon_5411x.h.

#### 7.40.2.49 __I uint32_t RESERVED16A

Definition at line 88 of file syscon_5411x.h.

#### 7.40.2.50 __I uint32_t RESERVED17[4]

Definition at line 91 of file syscon_5411x.h.

#### 7.40.2.51 __I uint32_t RESERVED18[31]

Definition at line 93 of file syscon_5411x.h.

#### 7.40.2.52 __I uint32_t RESERVED19[2]

Definition at line 96 of file syscon_5411x.h.

#### 7.40.2.53 __I uint32_t RESERVED2[1]

Definition at line 53 of file syscon_5411x.h.

#### 7.40.2.54 __I uint32_t RESERVED20

Definition at line 100 of file syscon_5411x.h.

#### 7.40.2.55 __I uint32_t RESERVED21

Definition at line 102 of file syscon_5411x.h.

#### 7.40.2.56 __I uint32_t RESERVED22[20]

Definition at line 105 of file syscon_5411x.h.

#### 7.40.2.57 __I uint32_t RESERVED23[2]

Definition at line 107 of file syscon_5411x.h.

#### 7.40.2.58 __I uint32_t RESERVED24

Definition at line 110 of file syscon_5411x.h.

#### 7.40.2.59 __I uint32_t RESERVED25

Definition at line 112 of file syscon_5411x.h.

#### 7.40.2.60 __I uint32_t RESERVED26[55]

Definition at line 114 of file syscon_5411x.h.

#### 7.40.2.61 __I uint32_t RESERVED27

Definition at line 116 of file syscon_5411x.h.

#### 7.40.2.62 __I uint32_t RESERVED28[28]

Definition at line 119 of file syscon_5411x.h.

#### 7.40.2.63 __I uint32_t RESERVED29[30]

Definition at line 125 of file syscon_5411x.h.

#### 7.40.2.64 __I uint32_t RESERVED3[28]

Definition at line 56 of file syscon_5411x.h.

#### 7.40.2.65 __I uint32_t RESERVED30[2]

Definition at line 127 of file syscon_5411x.h.

#### 7.40.2.66 __I uint32_t RESERVED31[2]

Definition at line 129 of file syscon_5411x.h.

#### 7.40.2.67 __I uint32_t RESERVED32[18]

Definition at line 131 of file syscon_5411x.h.

#### 7.40.2.68 __I uint32_t RESERVED33[6]

Definition at line 133 of file syscon_5411x.h.

#### 7.40.2.69 __I uint32_t RESERVED34[6]

Definition at line 135 of file syscon_5411x.h.

#### 7.40.2.70 __I uint32_t RESERVED35[78]

Definition at line 137 of file syscon_5411x.h.

#### 7.40.2.71 __I uint32_t RESERVED36[381]

Definition at line 142 of file syscon_5411x.h.

#### 7.40.2.72 __I uint32_t RESERVED37[123]

Definition at line 144 of file syscon_5411x.h.

#### 7.40.2.73 __I uint32_t RESERVED4[2]

Definition at line 58 of file syscon_5411x.h.

#### 7.40.2.74 __I uint32_t RESERVED5[10]

Definition at line 60 of file syscon_5411x.h.

#### 7.40.2.75 __I uint32_t RESERVED6[6]

Definition at line 62 of file syscon_5411x.h.

#### 7.40.2.76 __I uint32_t RESERVED7[6]

Definition at line 64 of file syscon_5411x.h.

#### 7.40.2.77 __I uint32_t RESERVED8[42]

Definition at line 66 of file syscon_5411x.h.

#### 7.40.2.78 __I uint32_t RESERVED9[3]

Definition at line 68 of file syscon_5411x.h.

#### 7.40.2.79 __IO uint32_t RTCOSCCTRL

RTC Oscillator control register

Definition at line 118 of file syscon_5411x.h.

#### 7.40.2.80 __IO uint32_t SPIFICLKDIV

SPIFI clock divider register

Definition at line 97 of file syscon_5411x.h.

#### 7.40.2.81 __IO uint32_t SPIFICLKSEL

SPIFI clock selection register

Definition at line 81 of file syscon_5411x.h.

#### 7.40.2.82 __IO uint32_t STARTERP[2]

Start logic wakeup enable register

Definition at line 132 of file syscon_5411x.h.

#### 7.40.2.83 __O uint32_t STARTERPCLR[2]

Start logic wakeup enable clear register

Definition at line 136 of file syscon_5411x.h.

#### 7.40.2.84 __O uint32_t STARTERPSET[2]

Start logic wakeup enable set register

Definition at line 134 of file syscon_5411x.h.

#### 7.40.2.85 __IO uint32_t SYSMEMREMAP

System Remap register

Definition at line 48 of file syscon_5411x.h.

#### 7.40.2.86 __IO uint32_t SYSPLLCLKSEL

System PLL Clk Selregister

Definition at line 79 of file syscon_5411x.h.

#### 7.40.2.87 __IO uint32_t SYSPLLCTRL

System PLL control register

Definition at line 120 of file syscon_5411x.h.

#### 7.40.2.88 __IO uint32_t SYSPLLNDEC

System PLL N-DEC register

Definition at line 122 of file syscon_5411x.h.

#### 7.40.2.89 __IO uint32_t SYSPLLPDEC

System PLL P-DEC register

Definition at line 123 of file syscon_5411x.h.

#### 7.40.2.90 __IO uint32_t SYSPLLSSCTRL[2]

System PLL Spread-Spectrum control register

Definition at line 124 of file syscon_5411x.h.

#### 7.40.2.91 __I uint32_t SYSPLLSTAT

System PLL status register

Definition at line 121 of file syscon_5411x.h.

#### 7.40.2.92 __IO uint32_t SYSRSTSTAT

System Reset Stat register

Definition at line 67 of file syscon_5411x.h.

#### 7.40.2.93 __IO uint32_t SYSTCKCAL

System Tick Calibration register

Definition at line 52 of file syscon_5411x.h.

#### 7.40.2.94 __IO uint32_t SYSTICKCLKDIV

Systick Clock divider register

Definition at line 92 of file syscon_5411x.h.

#### 7.40.2.95 __IO uint32_t USBCLKCTRL

USB Clock control register

Definition at line 108 of file syscon_5411x.h.

#### 7.40.2.96 __IO uint32_t USBCLKDIV

USB Clock divider register

Definition at line 99 of file syscon_5411x.h.

#### 7.40.2.97 __IO uint32_t USBCLKSEL

USB Async Clk Sel register

Definition at line 83 of file syscon_5411x.h.

#### 7.40.2.98 __IO uint32_t USBCLKSTAT

USB Clock Status register

Definition at line 109 of file syscon_5411x.h.

#### 7.40.2.99 __IO uint32_t WDTOSCCTRL

Watchdog Oscillator control

Definition at line 117 of file syscon_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/syscon_5411x.h

## 7.41 LPC_TIMER_T Struct Reference

### 7.41.1 Detailed Description

32-bit Standard timer register block structure

Definition at line 47 of file timer_5411x.h.

```
#include "timer_5411x.h"
```

### Data Fields

- __IO uint32_t IR
- __IO uint32_t TCR
- __IO uint32_t TC
- __IO uint32_t PR
- __IO uint32_t PC
- __IO uint32_t MCR
- __IO uint32_t MR [4]
- __IO uint32_t CCR
- __IO uint32_t CR [4]
- __IO uint32_t EMR
- __I uint32_t RESERVED0 [12]
- __IO uint32_t CTCR
- __IO uint32_t PWMC

### 7.41.2 Field Documentation

#### 7.41.2.1 __IO uint32_t CCR

Capture Control Register. The CCR controls which edges of the capture inputs are used to load the Capture Registers and whether or not an interrupt is generated when a capture takes place.

Definition at line 55 of file timer_5411x.h.

#### 7.41.2.2 __IO uint32_t CR[4]

Capture Register. CR is loaded with the value of TC when there is an event on the CAPn.0 input.

Definition at line 56 of file timer_5411x.h.

#### 7.41.2.3 __IO uint32_t CTCR

Count Control Register. The CTCR selects between Timer and Counter mode, and in Counter mode selects the signal and edge(s) for counting.

Definition at line 59 of file timer_5411x.h.

#### 7.41.2.4 __IO uint32_t EMR

External Match Register. The EMR controls the external match pins MATn.0-3 (MAT0.0-3 and MAT1.0-3 respectively).

Definition at line 57 of file timer_5411x.h.

#### 7.41.2.5 __IO uint32_t IR

< TIMERn Structure Interrupt Register. The IR can be written to clear interrupts. The IR can be read to identify which of eight possible interrupt sources are pending.

Definition at line 48 of file timer_5411x.h.

#### 7.41.2.6 __IO uint32_t MCR

Match Control Register. The MCR is used to control if an interrupt is generated and if the TC is reset when a Match occurs.

Definition at line 53 of file timer_5411x.h.

#### 7.41.2.7 __IO uint32_t MR[4]

Match Register. MR can be enabled through the MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt every time MR matches the TC.

Definition at line 54 of file timer_5411x.h.

#### 7.41.2.8 __IO uint32_t PC

Prescale Counter. The 32 bit PC is a counter which is incremented to the value stored in PR. When the value in PR is reached, the TC is incremented and the PC is cleared. The PC is observable and controllable through the bus interface.

Definition at line 52 of file timer_5411x.h.

#### 7.41.2.9 __IO uint32_t PR

Prescale Register. The Prescale Counter (below) is equal to this value, the next clock increments the TC and clears the PC.

Definition at line 51 of file timer_5411x.h.

#### 7.41.2.10 __IO uint32_t PWMC

Definition at line 60 of file timer_5411x.h.

#### 7.41.2.11 __I uint32_t RESERVED0[12]

Definition at line 58 of file timer_5411x.h.

#### 7.41.2.12 __IO uint32_t TC

Timer Counter. The 32 bit TC is incremented every PR+1 cycles of PCLK. The TC is controlled through the TCR.

Definition at line 50 of file timer_5411x.h.

#### 7.41.2.13 __IO uint32_t TCR

Timer Control Register. The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.

Definition at line 49 of file timer_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/timer_5411x.h

## 7.42 LPC_USART_T Struct Reference

### 7.42.1 Detailed Description

UART Registers.

Definition at line 53 of file uart_5411x.h.

```
#include "uart_5411x.h"
```

### Data Fields

- __IO uint32_t CFG
- __IO uint32_t CTL
- __IO uint32_t STAT
- __IO uint32_t INTENSET
- __O uint32_t INTENCLR
- __IO uint32_t BRG
- __I uint32_t INTSTAT
- __IO uint32_t OSR
- __IO uint32_t ADDR
- __IO uint32_t FIFOCFG
- __IO uint32_t FIFOSTAT
- __IO uint32_t FIFOTRIG
- __IO uint32_t FIFOINTENSET
- __IO uint32_t FIFOINTENCLR
- __IO uint32_t FIFOINTSTAT
- __O uint32_t FIFOWR
- __I uint32_t FIFORD
- __I uint32_t FIFORDNOPOP
- __IO uint32_t PSELID
- __I uint32_t PID

### 7.42.2 Field Documentation

#### 7.42.2.1 __IO uint32_t ADDR

Offset: 0x02C Address register (for automatic address matching)

Definition at line 63 of file uart_5411x.h.

#### 7.42.2.2 __IO uint32_t BRG

Offset: 0x020 Baud Rate Generator register

Definition at line 60 of file uart_5411x.h.

#### 7.42.2.3 __IO uint32_t CFG

Offset: 0x000 Configuration register

Definition at line 54 of file uart_5411x.h.

#### 7.42.2.4 __IO uint32_t CTL

Offset: 0x004 Control register

Definition at line 55 of file uart_5411x.h.

#### 7.42.2.5 __IO uint32_t FIFOCFG

Offset: 0xE00 FIFO Configuration register

Definition at line 67 of file uart_5411x.h.

#### 7.42.2.6 __IO uint32_t FIFOINTENCLR

Offset: 0xE14 FIFO Interrupt enable CLEAR register

Definition at line 72 of file uart_5411x.h.

#### 7.42.2.7 __IO uint32_t FIFOINTENSET

Offset: 0xE10 FIFO Interrupt enable SET register

Definition at line 71 of file uart_5411x.h.

#### 7.42.2.8 __IO uint32_t FIFOINTSTAT

Offset: 0xE18 FIFO Interrupt Status register

Definition at line 73 of file uart_5411x.h.

#### 7.42.2.9 __I uint32_t FIFORD

Offset: 0xE30 FIFO Data read register

Definition at line 77 of file uart_5411x.h.

#### 7.42.2.10 __I uint32_t FIFORDNOPOP

Offset: 0xE40 FIFO Data peek (read without popping out of queue) register

Definition at line 79 of file uart_5411x.h.

#### 7.42.2.11 __IO uint32_t FIFOSTAT

Offset: 0xE04 FIFO Status register

Definition at line 68 of file uart_5411x.h.

#### 7.42.2.12 __IO uint32_t FIFOTRIG

Offset: 0xE08 FIFO Trigger level register

Definition at line 69 of file uart_5411x.h.

#### 7.42.2.13 __O uint32_t FIFOWR

Offset: 0xE20 FIFO Data write register

Definition at line 75 of file uart_5411x.h.

#### 7.42.2.14 __O uint32_t INTENCLR

Offset: 0x010 Interrupt Enable Clear register

Definition at line 58 of file uart_5411x.h.

#### 7.42.2.15 __IO uint32_t INTENSET

Offset: 0x00C Interrupt Enable Read and Set register

Definition at line 57 of file uart_5411x.h.

#### 7.42.2.16 __I uint32_t INTSTAT

Offset: 0x024 Interrupt Status register

Definition at line 61 of file uart_5411x.h.

#### 7.42.2.17 __IO uint32_t OSR

Offset: 0x028 Oversampling register

Definition at line 62 of file uart_5411x.h.

#### 7.42.2.18 __I uint32_t PID

Offset: 0xFFC Module identification register

Definition at line 84 of file uart_5411x.h.

#### 7.42.2.19 __IO uint32_t PSELID

Offset: 0xFF8 Peripheral select/identification register

Definition at line 83 of file uart_5411x.h.

#### 7.42.2.20 __IO uint32_t STAT

Offset: 0x008 Status register

Definition at line 56 of file uart_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/uart_5411x.h

## 7.43 LPC_UTICK_T Struct Reference

### 7.43.1 Detailed Description

Micro Tick register block structure.

Definition at line 47 of file utick_5411x.h.

```
#include "utick_5411x.h"
```

### Data Fields

- __IO uint32_t CTRL
- __IO uint32_t STATUS

### 7.43.2 Field Documentation

#### 7.43.2.1 __IO uint32_t CTRL

UTick Control register

Definition at line 48 of file utick_5411x.h.

#### 7.43.2.2 __IO uint32_t STATUS

UTick Status register

Definition at line 49 of file utick_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/utick_5411x.h

## 7.44 LPC_WWDT_T Struct Reference

### 7.44.1 Detailed Description

Windowed Watchdog register block structure.

Definition at line 47 of file wwdt_5411x.h.

```
#include "wwdt_5411x.h"
```

### Data Fields

- __IO uint32_t MOD
- __IO uint32_t TC
- __O uint32_t FEED
- __I uint32_t TV
- __I uint32_t RESERVED0
- __IO uint32_t WARNINT
- __IO uint32_t WINDOW

### 7.44.2 Field Documentation

#### 7.44.2.1 __O uint32_t FEED

Watchdog feed sequence register. Writing 0xAA followed by 0x55 to this register reloads the Watchdog timer with the value contained in WDTC.

Definition at line 50 of file wwdt_5411x.h.

#### 7.44.2.2 __IO uint32_t MOD

< WWDT Structure Watchdog mode register. This register contains the basic mode and status of the Watchdog Timer.

Definition at line 48 of file wwdt_5411x.h.

#### 7.44.2.3 __I uint32_t RESERVED0

Definition at line 52 of file wwdt_5411x.h.

#### 7.44.2.4 __IO uint32_t TC

Watchdog timer constant register. This register determines the time-out value.

Definition at line 49 of file wwdt_5411x.h.

#### 7.44.2.5 __I uint32_t TV

Watchdog timer value register. This register reads out the current value of the Watchdog timer.

Definition at line 51 of file wwdt_5411x.h.

#### 7.44.2.6 __IO uint32_t WARNINT

Watchdog warning interrupt register. This register contains the Watchdog warning interrupt compare value.

Definition at line 53 of file wwdt_5411x.h.

#### 7.44.2.7 __IO uint32_t WINDOW

Watchdog timer window register. This register contains the Watchdog window value.

Definition at line 54 of file wwdt_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/wwdt_5411x.h

## 7.45 NVIC_Type Struct Reference

### 7.45.1 Detailed Description

Structure type to access the Nested Vectored Interrupt Controller (NVIC).

Definition at line 291 of file core_cm0plus.h.

```
#include "core_cm0plus.h"
```

### Data Fields

- __IO uint32_t ISER [1]
- uint32_t RESERVED0 [31]
- __IO uint32_t ICER [1]
- uint32_t RSERVED1 [31]
- __IO uint32_t ISPR [1]
- uint32_t RESERVED2 [31]
- __IO uint32_t ICPR [1]
- uint32_t RESERVED3 [31]
- uint32_t RESERVED4 [64]
- __IO uint32_t IP [8]
- __IO uint32_t IABR [8]
- __IO uint8_t IP [240]
- uint32_t RESERVED5 [644]
- __O uint32_t STIR

## 7.45.2 Field Documentation

### 7.45.2.1 __IO uint32_t IABR[8]

Offset: 0x200 (R/W) Interrupt Active bit Register

Definition at line 346 of file core_cm4.h.

### 7.45.2.2 __IO uint32_t ICER

Offset: 0x080 (R/W) Interrupt Clear Enable Register

Definition at line 295 of file core_cm0plus.h.

### 7.45.2.3 __IO uint32_t ICPR

Offset: 0x180 (R/W) Interrupt Clear Pending Register

Definition at line 299 of file core_cm0plus.h.

### 7.45.2.4 __IO uint32_t IP[8]

Offset: 0x300 (R/W) Interrupt Priority Register

Definition at line 302 of file core_cm0plus.h.

### 7.45.2.5 __IO uint8_t IP[240]

Offset: 0x300 (R/W) Interrupt Priority Register (8Bit wide)

Definition at line 348 of file core_cm4.h.

### 7.45.2.6 __IO uint32_t ISER

Offset: 0x000 (R/W) Interrupt Set Enable Register

Definition at line 293 of file core_cm0plus.h.

#### 7.45.2.7 __IO uint32_t ISPR

Offset: 0x100 (R/W) Interrupt Set Pending Register

Definition at line 297 of file core_cm0plus.h.

#### 7.45.2.8 uint32_t RESERVED0

Definition at line 294 of file core_cm0plus.h.

#### 7.45.2.9 uint32_t RESERVED2

Definition at line 298 of file core_cm0plus.h.

#### 7.45.2.10 uint32_t RESERVED3

Definition at line 300 of file core_cm0plus.h.

#### 7.45.2.11 uint32_t RESERVED4

Definition at line 301 of file core_cm0plus.h.

#### 7.45.2.12 uint32_t RESERVED5[644]

Definition at line 349 of file core_cm4.h.

#### 7.45.2.13 uint32_t RSERVED1

Definition at line 296 of file core_cm0plus.h.

#### 7.45.2.14 __O uint32_t STIR

Offset: 0xE00 ( /W) Software Trigger Interrupt Register

Definition at line 350 of file core_cm4.h.

The documentation for this struct was generated from the following files:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h
- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.46 PINMUX_GRP_T Struct Reference

### 7.46.1 Detailed Description

Array of IOCON pin definitions passed to Chip_IOCON_SetPinMuxing() must be in this format.

Definition at line 54 of file iocon_5411x.h.

```
#include "iocon_5411x.h"
```

## Data Fields

- uint32_t port: 8
- uint32_t pin: 8
- uint32_t modefunc: 16

### 7.46.2 Field Documentation

#### 7.46.2.1 uint32_t modefunc

Definition at line 57 of file iocon_5411x.h.

#### 7.46.2.2 uint32_t pin

Definition at line 56 of file iocon_5411x.h.

#### 7.46.2.3 uint32_t port

Definition at line 55 of file iocon_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/iocon_5411x.h

## 7.47 PINTABLE_T Struct Reference

### 7.47.1 Detailed Description

LPC5411X Pin table structure used for enhanced boot block support.

Definition at line 84 of file pintable_5411x.h.

```
#include "pintable_5411x.h"
```

## Data Fields

- uint32_t marker
- uint8_t img_type
- uint8_t ifSel
- uint8_t hostIrqPortPin
- uint8_t hostMisoPortPin
- uint8_t hostMosiPortPin
- uint8_t hostSselPortPin
- uint8_t hostSckPortPin
- uint8_t xorVal
- uint32_t crc32_len
- uint32_t crc32_val
- uint32_t version

### 7.47.2 Field Documentation

#### 7.47.2.1 uint32_t crc32_len

Definition at line 110 of file pintable_5411x.h.

#### 7.47.2.2 uint32_t crc32_val

Definition at line 111 of file pintable_5411x.h.

#### 7.47.2.3 uint8_t hostIrqPortPin

Definition at line 98 of file pintable_5411x.h.

#### 7.47.2.4 uint8_t hostMisoPortPin

Definition at line 100 of file pintable_5411x.h.

#### 7.47.2.5 uint8_t hostMosiPortPin

Definition at line 102 of file pintable_5411x.h.

#### 7.47.2.6 uint8_t hostSckPortPin

Definition at line 106 of file pintable_5411x.h.

#### 7.47.2.7 uint8_t hostSselPortPin

Definition at line 104 of file pintable_5411x.h.

#### 7.47.2.8 uint8_t ifSel

Definition at line 96 of file pintable_5411x.h.

#### 7.47.2.9 uint8_t img_type

Definition at line 93 of file pintable_5411x.h.

#### 7.47.2.10 uint32_t marker

Definition at line 86 of file pintable_5411x.h.

#### 7.47.2.11 uint32_t version

Definition at line 113 of file pintable_5411x.h.

#### 7.47.2.12 uint8_t xorVal

Definition at line 108 of file pintable_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pintable_5411x.h

## 7.48 PLL_CONFIG_T Struct Reference

### 7.48.1 Detailed Description

PLL configuration structure This structure can be used to configure the settings for a PLL setup structure. Fill in the desired configuration for the PLL and call the PLL setup function to fill in a PLL setup structure.

Definition at line 226 of file pll_5411x.h.

```
#include "pll_5411x.h"
```

### Data Fields

- uint32_t desiredRate
- uint32_t InputRate
- uint32_t flags
- SS_PROGMODFM_T ss_mf
- SS_PROGMODDP_T ss_mr
- SS_MODWVCTRL_T ss_mc
- bool mfDither

### 7.48.2 Field Documentation

#### 7.48.2.1 uint32_t desiredRate

Desired PLL rate in Hz

Definition at line 227 of file pll_5411x.h.

#### 7.48.2.2 uint32_t flags

PLL configuration flags, Or'ed value of PLL_CONFIGFLAG_* definitions

Definition at line 229 of file pll_5411x.h.

#### 7.48.2.3 uint32_t InputRate

PLL input clock in Hz, only used if PLL_CONFIGFLAG_USEINRATE flag is set

Definition at line 228 of file pll_5411x.h.

#### 7.48.2.4 bool mfDither

false for fixed modulation frequency or true for dithering, only applicable when not using PLL_CONFIGFLAG_FORCENOFRACT flag

Definition at line 233 of file pll_5411x.h.

#### 7.48.2.5 SS_MODWVCTRL_T ss_mc

SS Modulation waveform control, only applicable when not using PLL_CONFIGFLAG_FORCENOFRACT flag

Definition at line 232 of file pll_5411x.h.

#### 7.48.2.6 SS_PROGMODFM_T ss_mf

SS Programmable modulation frequency, only applicable when not using PLL_CONFIGFLAG_FORCENOFRACT flag

Definition at line 230 of file pll_5411x.h.

#### 7.48.2.7 SS_PROGMODDP_T ss_mr

SS Programmable frequency modulation depth, only applicable when not using PLL_CONFIGFLAG_FORCENO↩FRACT flag

Definition at line 231 of file pll_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pll_5411x.h

## 7.49 PLL_SETUP_T Struct Reference

### 7.49.1 Detailed Description

PLL setup structure This structure can be used to pre-build a PLL setup configuration at run-time and quickly set the PLL to the configuration. It can be populated with the PLL setup function. If powering up or waiting for PLL lock, the PLL input clock source should be configured prior to PLL setup.

Definition at line 250 of file pll_5411x.h.

```
#include "pll_5411x.h"
```

### Data Fields

- uint32_t SYSPLLCTRL
- uint32_t SYSPLLNDEC
- uint32_t SYSPLLPDEC
- uint32_t SYSPLLSSCTRL [2]
- uint32_t pllRate
- uint32_t flags

### 7.49.2 Field Documentation

#### 7.49.2.1 uint32_t flags

PLL setup flags, Or'ed value of PLL_SETUPFLAG_∗ definitions

Definition at line 256 of file pll_5411x.h.

#### 7.49.2.2 uint32_t pllRate

Acutal PLL rate

Definition at line 255 of file pll_5411x.h.

#### 7.49.2.3 uint32_t SYSPLLCTRL

PLL control register

Definition at line 251 of file pll_5411x.h.

#### 7.49.2.4 uint32_t SYSPLLNDEC

PLL NDEC register

Definition at line 252 of file pll_5411x.h.

#### 7.49.2.5 uint32_t SYSPLLPDEC

PLL PDEC register

Definition at line 253 of file pll_5411x.h.

#### 7.49.2.6 uint32_t SYSPLLSSCTRL[2]

PLL SSCTL registers

Definition at line 254 of file pll_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pll_5411x.h

## 7.50 REQUEST_TYPE Union Reference

### 7.50.1 Detailed Description

Union of _BM_T struct and 8 bit byte.

Definition at line 141 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t B
- BM_T BM

### 7.50.2 Field Documentation

#### 7.50.2.1 uint8_t B

byte wide access memeber

Definition at line 143 of file usbd.h.

#### 7.50.2.2 BM_T BM

bitfield structure access memeber

Definition at line 144 of file usbd.h.

The documentation for this union was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

# 7.51 RINGBUFF_T Struct Reference

## 7.51.1 Detailed Description

Ring buffer structure.

Definition at line 46 of file ring_buffer.h.

```
#include "ring_buffer.h"
```

## Data Fields

- void ∗ data
- int count
- int itemSz
- uint32_t head
- uint32_t tail
- void ∗(∗ copy )(void ∗dst, const void ∗src, uint32_t len)

## 7.51.2 Field Documentation

### 7.51.2.1 void∗(∗ copy) (void ∗dst, const void ∗src, uint32_t len)

Definition at line 52 of file ring_buffer.h.

### 7.51.2.2 int count

Definition at line 48 of file ring_buffer.h.

### 7.51.2.3 void∗ data

Definition at line 47 of file ring_buffer.h.

### 7.51.2.4 uint32_t head

Definition at line 50 of file ring_buffer.h.

### 7.51.2.5 int itemSz

Definition at line 49 of file ring_buffer.h.

### 7.51.2.6 uint32_t tail

Definition at line 51 of file ring_buffer.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/ring_buffer.h

## 7.52 SCB_Type Struct Reference

### 7.52.1 Detailed Description

Structure type to access the System Control Block (SCB).

Definition at line 316 of file core_cm0plus.h.

```
#include "core_cm0plus.h"
```

### Data Fields

- __I uint32_t CPUID
- __IO uint32_t ICSR
- uint32_t RESERVED0
- __IO uint32_t AIRCR
- __IO uint32_t SCR
- __IO uint32_t CCR
- uint32_t RESERVED1
- __IO uint32_t SHP [2]
- __IO uint32_t SHCSR
- __IO uint32_t VTOR
- __IO uint8_t SHP [12]
- __IO uint32_t CFSR
- __IO uint32_t HFSR
- __IO uint32_t DFSR
- __IO uint32_t MMFAR
- __IO uint32_t BFAR
- __IO uint32_t AFSR
- __I uint32_t PFR [2]
- __I uint32_t DFR
- __I uint32_t ADR
- __I uint32_t MMFR [4]
- __I uint32_t ISAR [5]
- __IO uint32_t CPACR

### 7.52.2 Field Documentation

#### 7.52.2.1 __I uint32_t ADR

Offset: 0x04C (R/ ) Auxiliary Feature Register

Definition at line 386 of file core_cm4.h.

#### 7.52.2.2 __IO uint32_t AFSR

Offset: 0x03C (R/W) Auxiliary Fault Status Register

Definition at line 383 of file core_cm4.h.

#### 7.52.2.3 __IO uint32_t AIRCR

Offset: 0x00C (R/W) Application Interrupt and Reset Control Register

Definition at line 325 of file core_cm0plus.h.

#### 7.52.2.4 __IO uint32_t BFAR

Offset: 0x038 (R/W) BusFault Address Register

Definition at line 382 of file core_cm4.h.

#### 7.52.2.5 __IO uint32_t CCR

Offset: 0x014 (R/W) Configuration Control Register

Definition at line 327 of file core_cm0plus.h.

#### 7.52.2.6 __IO uint32_t CFSR

Offset: 0x028 (R/W) Configurable Fault Status Register

Definition at line 378 of file core_cm4.h.

#### 7.52.2.7 __IO uint32_t CPACR

Offset: 0x088 (R/W) Coprocessor Access Control Register

Definition at line 390 of file core_cm4.h.

#### 7.52.2.8 __I uint32_t CPUID

Offset: 0x000 (R/ ) CPUID Base Register

Definition at line 318 of file core_cm0plus.h.

#### 7.52.2.9 __I uint32_t DFR

Offset: 0x048 (R/ ) Debug Feature Register

Definition at line 385 of file core_cm4.h.

#### 7.52.2.10 __IO uint32_t DFSR

Offset: 0x030 (R/W) Debug Fault Status Register

Definition at line 380 of file core_cm4.h.

#### 7.52.2.11 __IO uint32_t HFSR

Offset: 0x02C (R/W) HardFault Status Register

Definition at line 379 of file core_cm4.h.

#### 7.52.2.12 __IO uint32_t ICSR

Offset: 0x004 (R/W) Interrupt Control and State Register

Definition at line 319 of file core_cm0plus.h.

#### 7.52.2.13 __I uint32_t ISAR[5]

Offset: 0x060 (R/ ) Instruction Set Attributes Register

Definition at line 388 of file core_cm4.h.

#### 7.52.2.14 __IO uint32_t MMFAR

Offset: 0x034 (R/W) MemManage Fault Address Register

Definition at line 381 of file core_cm4.h.

#### 7.52.2.15 __I uint32_t MMFR[4]

Offset: 0x050 (R/ ) Memory Model Feature Register

Definition at line 387 of file core_cm4.h.

#### 7.52.2.16 __I uint32_t PFR[2]

Offset: 0x040 (R/ ) Processor Feature Register

Definition at line 384 of file core_cm4.h.

#### 7.52.2.17 uint32_t RESERVED0

Definition at line 323 of file core_cm0plus.h.

#### 7.52.2.18 uint32_t RESERVED1

Definition at line 328 of file core_cm0plus.h.

#### 7.52.2.19 __IO uint32_t SCR

Offset: 0x010 (R/W) System Control Register

Definition at line 326 of file core_cm0plus.h.

#### 7.52.2.20 __IO uint32_t SHCSR

Offset: 0x024 (R/W) System Handler Control and State Register

Definition at line 330 of file core_cm0plus.h.

#### 7.52.2.21 __IO uint32_t SHP[2]

Offset: 0x01C (R/W) System Handlers Priority Registers. [0] is RESERVED

Definition at line 329 of file core_cm0plus.h.

#### 7.52.2.22 __IO uint8_t SHP[12]

Offset: 0x018 (R/W) System Handlers Priority Registers (4-7, 8-11, 12-15)

Definition at line 376 of file core_cm4.h.

#### 7.52.2.23 __IO uint32_t VTOR

Offset: 0x008 (R/W) Vector Table Offset Register

Definition at line 372 of file core_cm4.h.

The documentation for this struct was generated from the following files:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h
- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.53 SCnSCB_Type Struct Reference

### 7.53.1 Detailed Description

Structure type to access the System Control and ID Register not in the SCB.

Definition at line 585 of file core_cm4.h.

```
#include "core_cm4.h"
```

### Data Fields

- uint32_t RESERVED0 [1]
- __I uint32_t ICTR
- __IO uint32_t ACTLR

### 7.53.2 Field Documentation

#### 7.53.2.1 __IO uint32_t ACTLR

Offset: 0x008 (R/W) Auxiliary Control Register

Definition at line 589 of file core_cm4.h.

#### 7.53.2.2 __I uint32_t ICTR

Offset: 0x004 (R/ ) Interrupt Controller Type Register

Definition at line 588 of file core_cm4.h.

#### 7.53.2.3 uint32_t RESERVED0[1]

Definition at line 587 of file core_cm4.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.54 SPI_CFGSETUP_T Struct Reference

### 7.54.1 Detailed Description

SPI configuration structure used for setting up master/slave mode, LSB or MSB first, and SPI mode in a single function call.

Definition at line 378 of file spi_common_5411x.h.

```
#include "spi_common_5411x.h"
```

## Data Fields

- uint32_t master: 8
- uint32_t lsbFirst: 8
- SPI_CLOCK_MODE_T mode: 8
- uint32_t reserved: 8

### 7.54.2 Field Documentation

#### 7.54.2.1 uint32_t lsbFirst

Definition at line 380 of file spi_common_5411x.h.

#### 7.54.2.2 uint32_t master

Definition at line 379 of file spi_common_5411x.h.

#### 7.54.2.3 SPI_CLOCK_MODE_T mode

Definition at line 381 of file spi_common_5411x.h.

#### 7.54.2.4 uint32_t reserved

Definition at line 382 of file spi_common_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spi_common_5411x.h

## 7.55 SPIM_DELAY_CONFIG_T Struct Reference

### 7.55.1 Detailed Description

SPI Delay Configure Struct.

**Note**

> All values are in SPI clocks.

Definition at line 70 of file spim_5411x.h.

```
#include "spim_5411x.h"
```

## Data Fields

- uint8_t PreDelay
- uint8_t PostDelay
- uint8_t FrameDelay
- uint8_t TransferDelay

## 7.55.2 Field Documentation

### 7.55.2.1 uint8_t FrameDelay

Additional Post-delay, minimum is 1 clock (SSEL asserted), 0 - 15

Definition at line 73 of file spim_5411x.h.

### 7.55.2.2 uint8_t PostDelay

Additional Pre-delay, minimum is 1 clock (SSEL asserted), 0 - 15

Definition at line 72 of file spim_5411x.h.

### 7.55.2.3 uint8_t PreDelay

Definition at line 71 of file spim_5411x.h.

### 7.55.2.4 uint8_t TransferDelay

Delay between frames (SSEL asserted), 0 - 15

Definition at line 74 of file spim_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spim_5411x.h

# 7.56 SPIM_XFER_T Struct Reference

## 7.56.1 Detailed Description

SPI Master transfer data context.

**Note**

> When structure member *dataWidth* $>$ 8; *txData* and *rxData* must be of type (uint16_t $*$) otherwise it should be of type (uint8_t $*$)
> Member *options* must be OR'd values from SPI_XFER_OPT_$*$ (see #SPI_XFER_OPT_EOF)

Definition at line 156 of file spim_5411x.h.

```
#include "spim_5411x.h"
```

## Data Fields

- int($*$ eventCB )(LPC_SPI_T $*$pSPI, SPIM_EVENT_T evt, struct SPIM_XFER $*$xfer)
- void $*$ txData
- void $*$ rxData
- uint16_t txCount
- uint16_t rxCount
- uint16_t dataWidth
- uint16_t sselNum
- uint32_t option
- void $*$ usrData

- uint16_t txIndex
- uint16_t rxIndex
- SPIM_XFER_STATE_T state

## 7.56.2 Field Documentation

### 7.56.2.1 uint16_t dataWidth

Width of the data [Valid values: 1 to 16]

Definition at line 163 of file spim_5411x.h.

### 7.56.2.2 int(∗ eventCB) (LPC_SPI_T ∗pSPI, SPIM_EVENT_T evt, struct SPIM_XFER ∗xfer)

Definition at line 158 of file spim_5411x.h.

### 7.56.2.3 uint32_t option

Dummy data to be transfered when rxCount > txCount ORed with #SPI_XFER_OPT_EOF (if this functionality is required)

Definition at line 165 of file spim_5411x.h.

### 7.56.2.4 uint16_t rxCount

Number of data to be received [Not in bytes]

Definition at line 162 of file spim_5411x.h.

### 7.56.2.5 void∗ rxData

Pointer to buffer where received data be stored

Definition at line 160 of file spim_5411x.h.

### 7.56.2.6 uint16_t rxIndex

Index of memory to which next received data be stored

Definition at line 169 of file spim_5411x.h.

### 7.56.2.7 uint16_t sselNum

Slave select number to be asserted when transfering data [Valid values: 0 to 3]

Definition at line 164 of file spim_5411x.h.

### 7.56.2.8 SPIM_XFER_STATE_T state

State of the transfer

Definition at line 170 of file spim_5411x.h.

#### 7.56.2.9 uint16_t txCount

Number of data to be transmitted [Not in bytes]

Definition at line 161 of file spim_5411x.h.

#### 7.56.2.10 void∗ txData

Pointer to SPI master transfer event callback functions Pointer to buffer having transmit data

Definition at line 159 of file spim_5411x.h.

#### 7.56.2.11 uint16_t txIndex

Index of the next item to be transfered [If this is same as txCount then tx is complete]

Definition at line 168 of file spim_5411x.h.

#### 7.56.2.12 void∗ usrData

User data associated with this Xfer structure

Definition at line 166 of file spim_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spim_5411x.h

## 7.57 SPIS_XFER_T Struct Reference

### 7.57.1 Detailed Description

Slave transfer data context

Definition at line 58 of file spis_5411x.h.

```
#include "spis_5411x.h"
```

### Data Fields

- int(∗ eventCB )(LPC_SPI_T ∗pSPI, SPIS_EVENT_T evt, struct SPIS_XFER ∗xfer)
- void ∗ txData
- void ∗ rxData
- uint16_t rxCount
- uint16_t txCount
- uint16_t txIndex
- uint16_t rxIndex
- uint16_t thresCount
- uint16_t dataWidth
- uint8_t sselNum
- uint32_t ss_count
- bool ss_state

## 7.57.2 Field Documentation

### 7.57.2.1 uint16_t dataWidth

Width of the data [Valid values: 1 to 16]

Definition at line 67 of file spis_5411x.h.

### 7.57.2.2 int(∗ eventCB) (LPC_SPI_T ∗pSPI, SPIS_EVENT_T evt, struct SPIS_XFER ∗xfer)

Pointer to SPI slave callback functions

Definition at line 59 of file spis_5411x.h.

### 7.57.2.3 uint16_t rxCount

Size of the pRXData buffer in items (not bytes), modified by driver

Definition at line 62 of file spis_5411x.h.

### 7.57.2.4 void∗ rxData

RX Data pointer

Definition at line 61 of file spis_5411x.h.

### 7.57.2.5 uint16_t rxIndex

Total items (not bytes) transmitted, modified by driver

Definition at line 65 of file spis_5411x.h.

### 7.57.2.6 uint32_t ss_count

Count of ssel events from the data, modified by driver

Definition at line 69 of file spis_5411x.h.

### 7.57.2.7 bool ss_state

State of slave-select true = asserted, false = de-asserted

Definition at line 70 of file spis_5411x.h.

### 7.57.2.8 uint8_t sselNum

Slave number assigned to this transfer, 0 - 3, modified by driver

Definition at line 68 of file spis_5411x.h.

### 7.57.2.9 uint16_t thresCount

Number of data to receive to trigger SPIS_EVENT_THRESHOLD event

Definition at line 66 of file spis_5411x.h.

#### 7.57.2.10 uint16_t txCount

Number of items (not bytes) to send in pTXData buffer, modified by driver

Definition at line 63 of file spis_5411x.h.

#### 7.57.2.11 void∗ txData

TX Data pointer

Definition at line 60 of file spis_5411x.h.

#### 7.57.2.12 uint16_t txIndex

Total items (not bytes) received, modified by driver

Definition at line 64 of file spis_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spis_5411x.h

## 7.58 SysTick_Type Struct Reference

### 7.58.1 Detailed Description

Structure type to access the System Timer (SysTick).

Definition at line 431 of file core_cm0plus.h.

```
#include "core_cm0plus.h"
```

### Data Fields

- __IO uint32_t CTRL
- __IO uint32_t LOAD
- __IO uint32_t VAL
- __I uint32_t CALIB

### 7.58.2 Field Documentation

#### 7.58.2.1 __I uint32_t CALIB

Offset: 0x00C (R/ ) SysTick Calibration Register

Definition at line 436 of file core_cm0plus.h.

#### 7.58.2.2 __IO uint32_t CTRL

Offset: 0x000 (R/W) SysTick Control and Status Register

Definition at line 433 of file core_cm0plus.h.

#### 7.58.2.3 __IO uint32_t LOAD

Offset: 0x004 (R/W) SysTick Reload Value Register

Definition at line 434 of file core_cm0plus.h.

#### 7.58.2.4 __IO uint32_t VAL

Offset: 0x008 (R/W) SysTick Current Value Register

Definition at line 435 of file core_cm0plus.h.

The documentation for this struct was generated from the following files:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h
- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.59 TPI_Type Struct Reference

### 7.59.1 Detailed Description

Structure type to access the Trace Port Interface Register (TPI).

Definition at line 919 of file core_cm4.h.

```
#include "core_cm4.h"
```

### Data Fields

- __IO uint32_t SSPSR
- __IO uint32_t CSPSR
- uint32_t RESERVED0 [2]
- __IO uint32_t ACPR
- uint32_t RESERVED1 [55]
- __IO uint32_t SPPR
- uint32_t RESERVED2 [131]
- __I uint32_t FFSR
- __IO uint32_t FFCR
- __I uint32_t FSCR
- uint32_t RESERVED3 [759]
- __I uint32_t TRIGGER
- __I uint32_t FIFO0
- __I uint32_t ITATBCTR2
- uint32_t RESERVED4 [1]
- __I uint32_t ITATBCTR0
- __I uint32_t FIFO1
- __IO uint32_t ITCTRL
- uint32_t RESERVED5 [39]
- __IO uint32_t CLAIMSET
- __IO uint32_t CLAIMCLR
- uint32_t RESERVED7 [8]
- __I uint32_t DEVID
- __I uint32_t DEVTYPE

## 7.59.2 Field Documentation

### 7.59.2.1 __IO uint32_t ACPR

Offset: 0x010 (R/W) Asynchronous Clock Prescaler Register

Definition at line 924 of file core_cm4.h.

### 7.59.2.2 __IO uint32_t CLAIMCLR

Offset: 0xFA4 (R/W) Claim tag clear

Definition at line 941 of file core_cm4.h.

### 7.59.2.3 __IO uint32_t CLAIMSET

Offset: 0xFA0 (R/W) Claim tag set

Definition at line 940 of file core_cm4.h.

### 7.59.2.4 __IO uint32_t CSPSR

Offset: 0x004 (R/W) Current Parallel Port Size Register

Definition at line 922 of file core_cm4.h.

### 7.59.2.5 __I uint32_t DEVID

Offset: 0xFC8 (R/ ) TPIU_DEVID

Definition at line 943 of file core_cm4.h.

### 7.59.2.6 __I uint32_t DEVTYPE

Offset: 0xFCC (R/ ) TPIU_DEVTYPE

Definition at line 944 of file core_cm4.h.

### 7.59.2.7 __IO uint32_t FFCR

Offset: 0x304 (R/W) Formatter and Flush Control Register

Definition at line 929 of file core_cm4.h.

### 7.59.2.8 __I uint32_t FFSR

Offset: 0x300 (R/ ) Formatter and Flush Status Register

Definition at line 928 of file core_cm4.h.

### 7.59.2.9 __I uint32_t FIFO0

Offset: 0xEEC (R/ ) Integration ETM Data

Definition at line 933 of file core_cm4.h.

#### 7.59.2.10 __I uint32_t FIFO1

Offset: 0xEFC (R/ ) Integration ITM Data

Definition at line 937 of file core_cm4.h.

#### 7.59.2.11 __I uint32_t FSCR

Offset: 0x308 (R/ ) Formatter Synchronization Counter Register

Definition at line 930 of file core_cm4.h.

#### 7.59.2.12 __I uint32_t ITATBCTR0

Offset: 0xEF8 (R/ ) ITATBCTR0

Definition at line 936 of file core_cm4.h.

#### 7.59.2.13 __I uint32_t ITATBCTR2

Offset: 0xEF0 (R/ ) ITATBCTR2

Definition at line 934 of file core_cm4.h.

#### 7.59.2.14 __IO uint32_t ITCTRL

Offset: 0xF00 (R/W) Integration Mode Control

Definition at line 938 of file core_cm4.h.

#### 7.59.2.15 uint32_t RESERVED0[2]

Definition at line 923 of file core_cm4.h.

#### 7.59.2.16 uint32_t RESERVED1[55]

Definition at line 925 of file core_cm4.h.

#### 7.59.2.17 uint32_t RESERVED2[131]

Definition at line 927 of file core_cm4.h.

#### 7.59.2.18 uint32_t RESERVED3[759]

Definition at line 931 of file core_cm4.h.

#### 7.59.2.19 uint32_t RESERVED4[1]

Definition at line 935 of file core_cm4.h.

#### 7.59.2.20 uint32_t RESERVED5[39]

Definition at line 939 of file core_cm4.h.

#### 7.59.2.21 uint32_t RESERVED7[8]

Definition at line 942 of file core_cm4.h.

#### 7.59.2.22 __IO uint32_t SPPR

Offset: 0x0F0 (R/W) Selected Pin Protocol Register

Definition at line 926 of file core_cm4.h.

#### 7.59.2.23 __IO uint32_t SSPSR

Offset: 0x000 (R/ ) Supported Parallel Port Size Register

Definition at line 921 of file core_cm4.h.

#### 7.59.2.24 __I uint32_t TRIGGER

Offset: 0xEE8 (R/ ) TRIGGER

Definition at line 932 of file core_cm4.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

## 7.60 UART_BAUD_T Struct Reference

### 7.60.1 Detailed Description

UART Baud rate calculation structure.

**Note**

> Use oversampling (*ovr*) value other than 16, only if the difference between the actual baud and desired baud has an unacceptable error percentage. Smaller *ovr* values can cause the sampling position within the data-bit less accurate an may potentially cause more noise errors or incorrect data set ovr to $<$ 10 only when there is no other higher values suitable.

Definition at line 211 of file uart_5411x.h.

```
#include "uart_5411x.h"
```

### Data Fields

- uint32_t clk
- uint32_t baud
- uint8_t ovr
- uint8_t mul
- uint16_t div

## 7.60.2 Field Documentation

### 7.60.2.1 uint32_t baud

IN: Required baud rate; OUT: Actual baud rate

Definition at line 213 of file uart_5411x.h.

### 7.60.2.2 uint32_t clk

IN: Base clock to fractional divider; OUT: "Base clock rate for UART"

Definition at line 212 of file uart_5411x.h.

### 7.60.2.3 uint16_t div

OUT: Integer divider to divide the "Base clock rate for UART"

Definition at line 216 of file uart_5411x.h.

### 7.60.2.4 uint8_t mul

IN: 0 - calculate MUL, 1 - do't calculate (*clk*) has UART base clock; OUT: MUL value to be set in FRG register

Definition at line 215 of file uart_5411x.h.

### 7.60.2.5 uint8_t ovr

IN: Number of desired over samples [0-auto detect or values 5 to 16]; OUT: Auto detected over samples [unchanged if IN is not 0]

Definition at line 214 of file uart_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/uart_5411x.h

# 7.61 UART_STATISTICS_T Struct Reference

## 7.61.1 Detailed Description

UART statistics structure.

**Note**

> Maintains current UART statistics.

Definition at line 224 of file uart_5411x.h.

```
#include "uart_5411x.h"
```

## Data Fields

- uint32_t interrupts
- uint32_t lvl_tx
- uint32_t lvl_rx

- uint32_t fifo_err_tx
- uint32_t fifo_err_rx
- uint32_t uart_cts
- uint32_t uart_break
- uint32_t uart_start
- uint32_t uart_err_frame
- uint32_t uart_err_parity
- uint32_t uart_err_rx_noise
- uint32_t uart_err_auto_baud

## 7.61.2 Field Documentation

### 7.61.2.1 uint32_t fifo_err_rx

count: FIFO receive errors

Definition at line 229 of file uart_5411x.h.

### 7.61.2.2 uint32_t fifo_err_tx

count: FIFO transmit errors

Definition at line 228 of file uart_5411x.h.

### 7.61.2.3 uint32_t interrupts

count: interrupts

Definition at line 225 of file uart_5411x.h.

### 7.61.2.4 uint32_t lvl_rx

count: receive interrupts

Definition at line 227 of file uart_5411x.h.

### 7.61.2.5 uint32_t lvl_tx

count: transmit interrupts

Definition at line 226 of file uart_5411x.h.

### 7.61.2.6 uint32_t uart_break

count: UART break

Definition at line 231 of file uart_5411x.h.

### 7.61.2.7 uint32_t uart_cts

count: UART CTS

Definition at line 230 of file uart_5411x.h.

#### 7.61.2.8 uint32_t uart_err_auto_baud

count: UART auto-baud errors

Definition at line 236 of file uart_5411x.h.

#### 7.61.2.9 uint32_t uart_err_frame

count: UART frame errors

Definition at line 233 of file uart_5411x.h.

#### 7.61.2.10 uint32_t uart_err_parity

count: UART parity errors

Definition at line 234 of file uart_5411x.h.

#### 7.61.2.11 uint32_t uart_err_rx_noise

count: UART receive noise errors

Definition at line 235 of file uart_5411x.h.

#### 7.61.2.12 uint32_t uart_start

count: UART RX starts

Definition at line 232 of file uart_5411x.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/uart_5411x.h

## 7.62 USB_COMMON_DESCRIPTOR Struct Reference

### 7.62.1 Detailed Description

USB Common Descriptor

Definition at line 664 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType

### 7.62.2 Field Documentation

#### 7.62.2.1 uint8_t bDescriptorType

Descriptor Type

Definition at line 667 of file usbd.h.

#### 7.62.2.2 uint8_t bLength

Size of this descriptor in bytes

Definition at line 666 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.63 USB_CONFIGURATION_DESCRIPTOR Struct Reference

### 7.63.1 Detailed Description

USB Standard Configuration Descriptor

Definition at line 437 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint16_t wTotalLength
- uint8_t bNumInterfaces
- uint8_t bConfigurationValue
- uint8_t iConfiguration
- uint8_t bmAttributes
- uint8_t bMaxPower

### 7.63.2 Field Documentation

#### 7.63.2.1 uint8_t bConfigurationValue

Value to use as an argument to the SetConfiguration() request to select this configuration.

Definition at line 447 of file usbd.h.

#### 7.63.2.2 uint8_t bDescriptorType

CONFIGURATION Descriptor Type

Definition at line 440 of file usbd.h.

#### 7.63.2.3 uint8_t bLength

Size of this descriptor in bytes

Definition at line 439 of file usbd.h.

#### 7.63.2.4 uint8_t bmAttributes

Configuration characteristics
D7: Reserved (set to one)

D6: Self-powered
D5: Remote Wakeup
D4...0: Reserved (reset to zero)
D7 is reserved and must be set to one for historical reasons.
A device configuration that uses power from the bus and a local source reports a non-zero value in bMaxPower to indicate the amount of bus power required and sets D6. The actual power source at runtime may be determined using the GetStatus(DEVICE) request (see USB 2.0 spec Section 9.4.5).
If a device configuration supports remote wakeup, D5 is set to one.

Definition at line 452 of file usbd.h.

#### 7.63.2.5 uint8_t bMaxPower

Maximum power consumption of the USB device from the bus in this specific configuration when the device is fully operational. Expressed in 2 mA units (i.e., 50 = 100 mA).
Note: A device configuration reports whether the configuration is bus-powered or selfpowered. Device status reports whether the device is currently self-powered. If a device is disconnected from its external power source, it updates device status to indicate that it is no longer self-powered.
A device may not increase its power draw from the bus, when it loses its external power source, beyond the amount reported by its configuration.
If a device can continue to operate when disconnected from its external power source, it continues to do so. If the device cannot continue to operate, it fails operations it can no longer support. The USB System Software may determine the cause of the failure by checking the status and noting the loss of the devices power source.

Definition at line 468 of file usbd.h.

#### 7.63.2.6 uint8_t bNumInterfaces

Number of interfaces supported by this configuration

Definition at line 446 of file usbd.h.

#### 7.63.2.7 uint8_t iConfiguration

Index of string descriptor describing this configuration

Definition at line 450 of file usbd.h.

#### 7.63.2.8 uint16_t wTotalLength

Total length of data returned for this configuration. Includes the combined length of all descriptors (configuration, interface, endpoint, and class- or vendor-specific) returned for this configuration.

Definition at line 441 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.64 USB_DEVICE_DESCRIPTOR Struct Reference

### 7.64.1 Detailed Description

USB Standard Device Descriptor

Definition at line 352 of file usbd.h.

```
#include "usbd.h"
```

## Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint16_t bcdUSB
- uint8_t bDeviceClass
- uint8_t bDeviceSubClass
- uint8_t bDeviceProtocol
- uint8_t bMaxPacketSize0
- uint16_t idVendor
- uint16_t idProduct
- uint16_t bcdDevice
- uint8_t iManufacturer
- uint8_t iProduct
- uint8_t iSerialNumber
- uint8_t bNumConfigurations

## 7.64.2 Field Documentation

### 7.64.2.1 uint16_t bcdDevice

Device release number in binary-coded decimal.

Definition at line 409 of file usbd.h.

### 7.64.2.2 uint16_t bcdUSB

BUSB Specification Release Number in Binary-Coded Decimal (i.e., 2.10 is 210H). This field identifies the release of the USB Specification with which the device and its descriptors are compliant.

Definition at line 356 of file usbd.h.

### 7.64.2.3 uint8_t bDescriptorType

DEVICE Descriptor Type.

Definition at line 355 of file usbd.h.

### 7.64.2.4 uint8_t bDeviceClass

Class code (assigned by the USB-IF). If this field is reset to zero, each interface within a configuration specifies its own class information and the various interfaces operate independently.
If this field is set to a value between 1 and FEH, the device supports different class specifications on different interfaces and the interfaces may not operate independently. This value identifies the class definition used for the aggregate interfaces.
If this field is set to FFH, the device class is vendor-specific.

Definition at line 362 of file usbd.h.

### 7.64.2.5 uint8_t bDeviceProtocol

Protocol code (assigned by the USB-IF). These codes are qualified by the value of the bDeviceClass and the b↩ DeviceSubClass fields. If a device supports class-specific protocols on a device basis as opposed to an interface basis, this code identifies the protocols that the device uses as defined by the specification of the device class.
If this field is reset to zero, the device does not use class-specific protocols on a device basis. However, it may use

classspecific protocols on an interface basis.
If this field is set to FFH, the device uses a vendor-specific protocol on a device basis.

Definition at line 386 of file usbd.h.

#### 7.64.2.6 uint8_t bDeviceSubClass

Subclass code (assigned by the USB-IF). These codes are qualified by the value of the bDeviceClass field.
If the bDeviceClass field is reset to zero, this field must also be reset to zero.
If the bDeviceClass field is not set to FFH, all values are reserved for assignment by the USB-IF.

Definition at line 377 of file usbd.h.

#### 7.64.2.7 uint8_t bLength

Size of this descriptor in bytes.

Definition at line 354 of file usbd.h.

#### 7.64.2.8 uint8_t bMaxPacketSize0

Maximum packet size for endpoint zero (only 8, 16, 32, or 64 are valid). For HS devices is fixed to 64.

Definition at line 402 of file usbd.h.

#### 7.64.2.9 uint8_t bNumConfigurations

Number of possible configurations.

Definition at line 415 of file usbd.h.

#### 7.64.2.10 uint16_t idProduct

Product ID (assigned by the manufacturer).

Definition at line 408 of file usbd.h.

#### 7.64.2.11 uint16_t idVendor

Vendor ID (assigned by the USB-IF).

Definition at line 407 of file usbd.h.

#### 7.64.2.12 uint8_t iManufacturer

Index of string descriptor describing manufacturer.

Definition at line 410 of file usbd.h.

#### 7.64.2.13 uint8_t iProduct

Index of string descriptor describing product.

Definition at line 411 of file usbd.h.

#### 7.64.2.14 uint8_t iSerialNumber

Index of string descriptor describing the devices serial number.

Definition at line 412 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.65 USB_DEVICE_QUALIFIER_DESCRIPTOR Struct Reference

### 7.65.1 Detailed Description

USB 2.0 Device Qualifier Descriptor

Definition at line 421 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint16_t bcdUSB
- uint8_t bDeviceClass
- uint8_t bDeviceSubClass
- uint8_t bDeviceProtocol
- uint8_t bMaxPacketSize0
- uint8_t bNumConfigurations
- uint8_t bReserved

### 7.65.2 Field Documentation

#### 7.65.2.1 uint16_t bcdUSB

USB specification version number (e.g., 0200H for V2.00)

Definition at line 425 of file usbd.h.

#### 7.65.2.2 uint8_t bDescriptorType

Device Qualifier Type

Definition at line 424 of file usbd.h.

#### 7.65.2.3 uint8_t bDeviceClass

Class Code

Definition at line 426 of file usbd.h.

#### 7.65.2.4 uint8_t bDeviceProtocol

Protocol Code

Definition at line 428 of file usbd.h.

#### 7.65.2.5 uint8_t bDeviceSubClass

SubClass Code

Definition at line 427 of file usbd.h.

#### 7.65.2.6 uint8_t bLength

Size of descriptor

Definition at line 423 of file usbd.h.

#### 7.65.2.7 uint8_t bMaxPacketSize0

Maximum packet size for other speed

Definition at line 429 of file usbd.h.

#### 7.65.2.8 uint8_t bNumConfigurations

Number of Other-speed Configurations

Definition at line 430 of file usbd.h.

#### 7.65.2.9 uint8_t bReserved

Reserved for future use, must be zero

Definition at line 431 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.66 USB_ENDPOINT_DESCRIPTOR Struct Reference

### 7.66.1 Detailed Description

USB Standard Endpoint Descriptor

Definition at line 571 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint8_t bEndpointAddress
- uint8_t bmAttributes
- uint16_t wMaxPacketSize
- uint8_t bInterval

## 7.66.2 Field Documentation

### 7.66.2.1 uint8_t bDescriptorType

ENDPOINT Descriptor Type

Definition at line 574 of file usbd.h.

### 7.66.2.2 uint8_t bEndpointAddress

The address of the endpoint on the USB device described by this descriptor. The address is encoded as follows:
Bit 3...0: The endpoint number
Bit 6...4: Reserved, reset to zero
Bit 7: Direction, ignored for control endpoints 0 = OUT endpoint 1 = IN endpoint.

**See also**

USBD_ENDPOINT_ADR_Type

Definition at line 575 of file usbd.h.

### 7.66.2.3 uint8_t bInterval

Interval for polling endpoint for data transfers. Expressed in frames or microframes depending on the device operating speed (i.e., either 1 millisecond or 125 s units).

- For full-/high-speed isochronous endpoints, this value must be in the range from 1 to 16. The bInterval value is used as the exponent for a $2^(bInterval - 1)$ value; e.g., a bInterval of 4 means a period of 8 ( $2^(4 - 1)$).
- For full-/low-speed interrupt endpoints, the value of this field may be from 1 to 255.
- For high-speed interrupt endpoints, the bInterval value is used as the exponent for a $2^(bInterval - 1)$ value; e.g., a bInterval of 4 means a period of 8 ( $2^(4 - 1)$) . This value must be from 1 to 16.
- For high-speed bulk/control OUT endpoints, the bInterval must specify the maximum NAK rate of the endpoint. A value of 0 indicates the endpoint never NAKs. Other values indicate at most 1 NAK each bInterval number of microframes. This value must be in the range from 0 to 255.
  Refer to Chapter 5 of USB 2.0 specification for more information.

Definition at line 627 of file usbd.h.

### 7.66.2.4 uint8_t bLength

Size of this descriptor in bytes

Definition at line 573 of file usbd.h.

### 7.66.2.5 uint8_t bmAttributes

This field describes the endpoints attributes when it is configured using the bConfigurationValue.
Bits 1..0: Transfer Type

- 00 = Control
- 01 = Isochronous
- 10 = Bulk

- 11 = Interrupt
  If not an isochronous endpoint, bits 5..2 are reserved and must be set to zero. If isochronous, they are defined as follows:
  Bits 3..2: Synchronization Type
- 00 = No Synchronization
- 01 = Asynchronous
- 10 = Adaptive
- 11 = Synchronous
  Bits 5..4: Usage Type
- 00 = Data endpoint
- 01 = Feedback endpoint
- 10 = Implicit feedback Data endpoint
- 11 = Reserved
  Refer to Chapter 5 of USB 2.0 specification for more information.
  All other bits are reserved and must be reset to zero. Reserved bits must be ignored by the host.

**See also**

USBD_EP_ATTR_Type

Definition at line 583 of file usbd.h.

#### 7.66.2.6 uint16_t wMaxPacketSize

Maximum packet size this endpoint is capable of sending or receiving when this configuration is selected.
For isochronous endpoints, this value is used to reserve the bus time in the schedule, required for the per-(micro)frame data payloads. The pipe may, on an ongoing basis, actually use less bandwidth than that reserved. The device reports, if necessary, the actual bandwidth used via its normal, non-USB defined mechanisms.
For all endpoints, bits 10..0 specify the maximum packet size (in bytes).
For high-speed isochronous and interrupt endpoints:
Bits 12..11 specify the number of additional transaction opportunities per microframe:

- 00 = None (1 transaction per microframe)
- 01 = 1 additional (2 per microframe)
- 10 = 2 additional (3 per microframe)
- 11 = Reserved
  Bits 15..13 are reserved and must be set to zero.

Definition at line 607 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

# 7.67 USB_IAD_DESCRIPTOR Struct Reference

## 7.67.1 Detailed Description

USB Standard Interface Association Descriptor

Definition at line 497 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint8_t bFirstInterface
- uint8_t bInterfaceCount
- uint8_t bFunctionClass
- uint8_t bFunctionSubClass
- uint8_t bFunctionProtocol
- uint8_t iFunction

## 7.67.2 Field Documentation

### 7.67.2.1 uint8_t bDescriptorType

INTERFACE ASSOCIATION Descriptor Type

Definition at line 500 of file usbd.h.

### 7.67.2.2 uint8_t bFirstInterface

Interface number of the first interface that is associated with this function.

Definition at line 501 of file usbd.h.

### 7.67.2.3 uint8_t bFunctionClass

Class code (assigned by USB-IF).
A value of zero is not allowed in this descriptor. If this field is FFH, the function class is vendorspecific. All other values are reserved for assignment by the USB-IF.

Definition at line 505 of file usbd.h.

### 7.67.2.4 uint8_t bFunctionProtocol

Protocol code (assigned by the USB).
These codes are qualified by the values of the bFunctionClass and bFunctionSubClass fields.

Definition at line 513 of file usbd.h.

### 7.67.2.5 uint8_t bFunctionSubClass

Subclass code (assigned by USB-IF).
If the bFunctionClass field is not set to FFH all values are reserved for assignment by the USBIF.

Definition at line 510 of file usbd.h.

#### 7.67.2.6 uint8_t bInterfaceCount

Number of contiguous interfaces that are associated with this function.

Definition at line 503 of file usbd.h.

#### 7.67.2.7 uint8_t bLength

Size of this descriptor in bytes

Definition at line 499 of file usbd.h.

#### 7.67.2.8 uint8_t iFunction

Index of string descriptor describing this function.

Definition at line 516 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.68 USB_INTERFACE_DESCRIPTOR Struct Reference

### 7.68.1 Detailed Description

USB Standard Interface Descriptor

Definition at line 522 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint8_t bInterfaceNumber
- uint8_t bAlternateSetting
- uint8_t bNumEndpoints
- uint8_t bInterfaceClass
- uint8_t bInterfaceSubClass
- uint8_t bInterfaceProtocol
- uint8_t iInterface

### 7.68.2 Field Documentation

#### 7.68.2.1 uint8_t bAlternateSetting

Value used to select this alternate setting for the interface identified in the prior field

Definition at line 530 of file usbd.h.

#### 7.68.2.2 uint8_t bDescriptorType

INTERFACE Descriptor Type

Definition at line 525 of file usbd.h.

#### 7.68.2.3 uint8_t bInterfaceClass

Class code (assigned by the USB-IF).
A value of zero is reserved for future standardization.
If this field is set to FFH, the interface class is vendor-specific.
All other values are reserved for assignment by the USB-IF.

Definition at line 536 of file usbd.h.

#### 7.68.2.4 uint8_t bInterfaceNumber

Number of this interface. Zero-based value identifying the index in the array of concurrent interfaces supported by this configuration.

Definition at line 526 of file usbd.h.

#### 7.68.2.5 uint8_t bInterfaceProtocol

Protocol code (assigned by the USB).
These codes are qualified by the value of the bInterfaceClass and the bInterfaceSubClass fields. If an interface supports class-specific requests, this code identifies the protocols that the device uses as defined by the specification of the device class.
If this field is reset to zero, the device does not use a class-specific protocol on this interface.
If this field is set to FFH, the device uses a vendor-specific protocol for this interface.

Definition at line 551 of file usbd.h.

#### 7.68.2.6 uint8_t bInterfaceSubClass

Subclass code (assigned by the USB-IF).
These codes are qualified by the value of the bInterfaceClass field.
If the bInterfaceClass field is reset to zero, this field must also be reset to zero.
If the bInterfaceClass field is not set to FFH, all values are reserved for assignment by the USB-IF.

Definition at line 543 of file usbd.h.

#### 7.68.2.7 uint8_t bLength

Size of this descriptor in bytes

Definition at line 524 of file usbd.h.

#### 7.68.2.8 uint8_t bNumEndpoints

Number of endpoints used by this interface (excluding endpoint zero). If this value is zero, this interface only uses the Default Control Pipe.

Definition at line 532 of file usbd.h.

#### 7.68.2.9 uint8_t iInterface

Index of string descriptor describing this interface

Definition at line 565 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.69 USB_OTHER_SPEED_CONFIGURATION Struct Reference

### 7.69.1 Detailed Description

USB Other Speed Configuration

Definition at line 673 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint16_t wTotalLength
- uint8_t bNumInterfaces
- uint8_t bConfigurationValue
- uint8_t IConfiguration
- uint8_t bmAttributes
- uint8_t bMaxPower

### 7.69.2 Field Documentation

#### 7.69.2.1 uint8_t bConfigurationValue

Value to use to select configuration

Definition at line 679 of file usbd.h.

#### 7.69.2.2 uint8_t bDescriptorType

Other_speed_Configuration Type

Definition at line 676 of file usbd.h.

#### 7.69.2.3 uint8_t bLength

Size of descriptor

Definition at line 675 of file usbd.h.

#### 7.69.2.4 uint8_t bmAttributes

Same as Configuration descriptor

Definition at line 681 of file usbd.h.

#### 7.69.2.5 uint8_t bMaxPower

Same as Configuration descriptor

Definition at line 682 of file usbd.h.

#### 7.69.2.6 uint8_t bNumInterfaces

Number of interfaces supported by this speed configuration

Definition at line 678 of file usbd.h.

#### 7.69.2.7 uint8_t IConfiguration

Index of string descriptor

Definition at line 680 of file usbd.h.

#### 7.69.2.8 uint16_t wTotalLength

Total length of data returned

Definition at line 677 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.70 USB_SETUP_PACKET Struct Reference

### 7.70.1 Detailed Description

USB Default Control Pipe Setup Packet

Definition at line 199 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- REQUEST_TYPE bmRequestType
- uint8_t bRequest
- WORD_BYTE wValue
- WORD_BYTE wIndex
- uint16_t wLength

### 7.70.2 Field Documentation

#### 7.70.2.1 REQUEST_TYPE bmRequestType

This bitmapped field identifies the characteristics of the specific request.

**See also**

_BM_T.

Definition at line 201 of file usbd.h.

#### 7.70.2.2 uint8_t bRequest

This field specifies the particular request. The Type bits in the bmRequestType field modify the meaning of this field.

**See also**

USBD_REQUEST.

Definition at line 204 of file usbd.h.

#### 7.70.2.3 WORD_BYTE wIndex

Used to pass a parameter to the device, specific to the request. The wIndex field is often used in requests to specify an endpoint or an interface.

Definition at line 211 of file usbd.h.

#### 7.70.2.4 uint16_t wLength

This field specifies the length of the data transferred during the second phase of the control transfer.

Definition at line 215 of file usbd.h.

#### 7.70.2.5 WORD_BYTE wValue

Used to pass a parameter to the device, specific to the request.

Definition at line 208 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.71 USB_STRING_DESCRIPTOR Struct Reference

### 7.71.1 Detailed Description

USB String Descriptor

Definition at line 654 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t bLength
- uint8_t bDescriptorType
- uint16_t bString

### 7.71.2 Field Documentation

#### 7.71.2.1 uint8_t bDescriptorType

STRING Descriptor Type

Definition at line 657 of file usbd.h.

#### 7.71.2.2 uint8_t bLength

Size of this descriptor in bytes

Definition at line 656 of file usbd.h.

#### 7.71.2.3 uint16_t bString

UNICODE encoded string

Definition at line 658 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.72 WB_T Struct Reference

### 7.72.1 Detailed Description

Structure to pack lower and upper byte to form 16 bit word.

Definition at line 78 of file usbd.h.

```
#include "usbd.h"
```

### Data Fields

- uint8_t L
- uint8_t H

### 7.72.2 Field Documentation

#### 7.72.2.1 uint8_t H

upper byte

Definition at line 81 of file usbd.h.

#### 7.72.2.2 uint8_t L

lower byte

Definition at line 80 of file usbd.h.

The documentation for this struct was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.73 WORD_BYTE Union Reference

### 7.73.1 Detailed Description

Union of _WB_T struct and 16 bit word.

Definition at line 87 of file usbd.h.

```
#include "usbd.h"
```

## Data Fields

- uint16_t W
- WB_T WB

### 7.73.2 Field Documentation

#### 7.73.2.1 uint16_t W

data member to do 16 bit access

Definition at line 89 of file usbd.h.

#### 7.73.2.2 WB_T WB

data member to do 8 bit access

Definition at line 90 of file usbd.h.

The documentation for this union was generated from the following file:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h

## 7.74 xPSR_Type Union Reference

### 7.74.1 Detailed Description

Union type to access the Special-Purpose Program Status Registers (xPSR).

Definition at line 242 of file core_cm0plus.h.

```
#include "core_cm0plus.h"
```

## Data Fields

- struct {
  uint32_t ISR:9
  uint32_t _reserved0:15
  uint32_t T:1
  uint32_t IT:2
  uint32_t Q:1
  uint32_t V:1
  uint32_t C:1
  uint32_t Z:1
  uint32_t N:1
  } b
- uint32_t w
- struct {
  uint32_t ISR:9
  uint32_t _reserved0:7
  uint32_t GE:4
  uint32_t _reserved1:4

```
        uint32_t T:1
        uint32_t IT:2
        uint32_t Q:1
                        uint32_t V:1
                        uint32_t C:1
                        uint32_t Z:1
                        uint32_t N:1
                      } b
```

## 7.74.2 Field Documentation

### 7.74.2.1 uint32_t _reserved0

bit: 9..23 Reserved

bit: 9..15 Reserved

Definition at line 248 of file core_cm0plus.h.

### 7.74.2.2 uint32_t _reserved1

bit: 20..23 Reserved

Definition at line 297 of file core_cm4.h.

### 7.74.2.3 struct { ... } b

Structure used for bit access

### 7.74.2.4 struct { ... } b

Structure used for bit access

### 7.74.2.5 uint32_t C

bit: 29 Carry condition code flag

Definition at line 258 of file core_cm0plus.h.

### 7.74.2.6 uint32_t GE

bit: 16..19 Greater than or Equal flags

Definition at line 296 of file core_cm4.h.

### 7.74.2.7 uint32_t ISR

bit: 0.. 8 Exception number

Definition at line 246 of file core_cm0plus.h.

### 7.74.2.8 uint32_t IT

bit: 25..26 saved IT state (read 0)

Definition at line 255 of file core_cm0plus.h.

#### 7.74.2.9 uint32_t N

bit: 31 Negative condition code flag

Definition at line 260 of file core_cm0plus.h.

#### 7.74.2.10 uint32_t Q

bit: 27 Saturation condition flag

Definition at line 256 of file core_cm0plus.h.

#### 7.74.2.11 uint32_t T

bit: 24 Thumb bit (read 0)

Definition at line 254 of file core_cm0plus.h.

#### 7.74.2.12 uint32_t V

bit: 28 Overflow condition code flag

Definition at line 257 of file core_cm0plus.h.

#### 7.74.2.13 uint32_t w

Type used for word access

Definition at line 262 of file core_cm0plus.h.

#### 7.74.2.14 uint32_t Z

bit: 30 Zero condition code flag

Definition at line 259 of file core_cm0plus.h.

The documentation for this union was generated from the following files:

- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h
- C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h

# Chapter 8

# File Documentation

## 8.1 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/adc_5411x.h File Reference

### Data Structures

- struct LPC_ADC_T

  *ADC register block structure.*

### Macros

- #define ADC_MAX_SAMPLE_RATE 80000000
- #define ADC_MAX_CHANNEL_NUM 12
- #define ADC_CR_CLKDIV_MASK (0xFF << 0)

  *ADC register support bitfields and mask.*
- #define ADC_CR_CLKDIV_BITPOS (0)
- #define ADC_CR_ASYNC_MODE (1 << 8)
- #define ADC_CR_RESOL(n) ((n) << 9)
- #define ADC_CR_LPWRMODEBIT (1 << 10)
- #define ADC_CR_BYPASS (1 << 11)
- #define ADC_CR_TSAMP(n) ((n) << 12)
- #define ADC_CR_CALMODEBIT (1 << 30)
- #define ADC_CR_BITACC(n) ((((n) & 0x1) << 9))
- #define ADC_CR_CLKDIV(n) ((((n) & 0xFF) << 0))
- #define ADC_SAMPLE_RATE_CONFIG_MASK (ADC_CR_CLKDIV(0xFF) | ADC_CR_BITACC(0x01))
- #define ADC_SEQ_CTRL_CHANNEL_EN(n) (1 << n)
- #define ADC_SEQ_CTRL_TRIGGER(n) ((n & 0x3f)<<12)
- #define ADC_SEQ_CTRL_HWTRIG_POLPOS (1 << 18)
- #define ADC_SEQ_CTRL_HWTRIG_SYNCBYPASS (1 << 19)
- #define ADC_SEQ_CTRL_START (1 << 26)
- #define ADC_SEQ_CTRL_BURST (1 << 27)
- #define ADC_SEQ_CTRL_SINGLESTEP (1 << 28)
- #define ADC_SEQ_CTRL_LOWPRIO (1 << 29)
- #define ADC_SEQ_CTRL_MODE_EOS (1 << 30)
- #define ADC_SEQ_CTRL_SEQ_ENA (1UL << 31)
- #define ADC_SEQ_GDAT_RESULT_MASK (0xFFF << 4)
- #define ADC_SEQ_GDAT_RESULT_BITPOS (4)
- #define ADC_SEQ_GDAT_THCMPRANGE_MASK (0x3 << 16)
- #define ADC_SEQ_GDAT_THCMPRANGE_BITPOS (16)
- #define ADC_SEQ_GDAT_THCMPCROSS_MASK (0x3 << 18)

- #define ADC_SEQ_GDAT_THCMPCROSS_BITPOS (18)
- #define ADC_SEQ_GDAT_CHAN_MASK (0xF << 26)
- #define ADC_SEQ_GDAT_CHAN_BITPOS (26)
- #define ADC_SEQ_GDAT_OVERRUN (1 << 30)
- #define ADC_SEQ_GDAT_DATAVALID (1UL << 31)
- #define ADC_DR_RESULT_BITPOS (4)
- #define ADC_DR_RESULT(n) ((((n) >> 4) & 0xFFF))
- #define ADC_DR_THCMPRANGE_MASK (0x3 << 16)
- #define ADC_DR_THCMPRANGE_BITPOS (16)
- #define ADC_DR_THCMPRANGE(n) (((n) >> ADC_DR_THCMPRANGE_BITPOS) & 0x3)
- #define ADC_DR_THCMPCROSS_MASK (0x3 << 18)
- #define ADC_DR_THCMPCROSS_BITPOS (18)
- #define ADC_DR_THCMPCROSS(n) (((n) >> ADC_DR_THCMPCROSS_BITPOS) & 0x3)
- #define ADC_DR_CHAN_MASK (0xF << 26)
- #define ADC_DR_CHAN_BITPOS (26)
- #define ADC_DR_CHANNEL(n) (((n) >> ADC_DR_CHAN_BITPOS) & 0xF)
- #define ADC_DR_OVERRUN (1 << 30)
- #define ADC_DR_DATAVALID (1UL << 31)
- #define ADC_DR_DONE(n) (((n) >> 31))
- #define ADC_THR_VAL_MASK (0xFFF << 4)
- #define ADC_THR_VAL_POS (4)
- #define ADC_THRSEL_CHAN_SEL_THR1(n) (1 << (n))
- #define ADC_INTEN_SEQA_ENABLE (1 << 0)
- #define ADC_INTEN_SEQB_ENABLE (1 << 1)
- #define ADC_INTEN_SEQN_ENABLE(seq) (1 << (seq))
- #define ADC_INTEN_OVRRUN_ENABLE (1 << 2)
- #define ADC_INTEN_CMP_DISBALE (0)
- #define ADC_INTEN_CMP_OUTSIDETH (1)
- #define ADC_INTEN_CMP_CROSSTH (2)
- #define ADC_INTEN_CMP_MASK (3)
- #define ADC_INTEN_CMP_ENABLE(isel, ch) (((isel) & ADC_INTEN_CMP_MASK) << ((2 ∗ (ch)) + 3))
- #define ADC_FLAGS_THCMP_MASK(ch) (1 << (ch))
- #define ADC_FLAGS_OVRRUN_MASK(ch) (1 << (12 + (ch)))
- #define ADC_FLAGS_SEQA_OVRRUN_MASK (1 << 24)
- #define ADC_FLAGS_SEQB_OVRRUN_MASK (1 << 25)
- #define ADC_FLAGS_SEQN_OVRRUN_MASK(seq) (1 << (24 + (seq)))
- #define ADC_FLAGS_SEQA_INT_MASK (1 << 28)
- #define ADC_FLAGS_SEQB_INT_MASK (1 << 29)
- #define ADC_FLAGS_SEQN_INT_MASK(seq) (1 << (28 + (seq)))
- #define ADC_FLAGS_THCMP_INT_MASK (1 << 30)
- #define ADC_FLAGS_OVRRUN_INT_MASK (1UL << 31)
- #define ADC_STARTUP_ENABLE (0x1 << 0)
- #define ADC_STARTUP_INIT (0x1 << 1)
- #define ADC_CALIB (0x1<<0)
- #define ADC_CALREQD (0x1<<1)

## Enumerations

- enum ADC_SEQ_IDX_T { ADC_SEQA_IDX = 0, ADC_SEQB_IDX }
- enum ADC_TSAMP_T {
  ADC_TSAMP_2CLK5 = 0, ADC_TSAMP_3CLK5, ADC_TSAMP_4CLK5, ADC_TSAMP_5CLK5,
  ADC_TSAMP_6CLK5, ADC_TSAMP_7CLK5, ADC_TSAMP_8CLK5, ADC_TSAMP_9CLK5 }

  *ADC sampling time bits 12, 13 and 14.*

- enum ADC_DR_THCMPRANGE_T { ADC_DR_THCMPRANGE_INRANGE, ADC_DR_THCMPRANGE_RESERVED, ADC_DR_THCMPRANGE_BELOW, ADC_DR_THCMPRANGE_ABOVE }
- enum ADC_DR_THCMPCROSS_T { ADC_DR_THCMPCROSS_NOCROSS, ADC_DR_THCMPCROSS_RESERVED, ADC_DR_THCMPCROSS_DOWNWARD, ADC_DR_THCMPCROSS_UPWARD }
- enum ADC_INTEN_THCMP_T { ADC_INTEN_THCMP_DISABLE, ADC_INTEN_THCMP_OUTSIDE, ADC_INTEN_THCMP_CROSSING }

## Functions

- void Chip_ADC_Init (LPC_ADC_T ∗pADC, uint32_t flags)

    *Initialize the ADC peripheral.*
- void Chip_ADC_DeInit (LPC_ADC_T ∗pADC)

    *Shutdown ADC.*
- __STATIC_INLINE void Chip_ADC_SetDivider (LPC_ADC_T ∗pADC, uint8_t div)

    *Set ADC divider.*
- void Chip_ADC_SetClockRate (LPC_ADC_T ∗pADC, uint32_t rate)

    *Set ADC clock rate.*
- __STATIC_INLINE uint8_t Chip_ADC_GetDivider (LPC_ADC_T ∗pADC)

    *Get ADC divider.*
- uint32_t Chip_ADC_Calibration (LPC_ADC_T ∗pADC)

    *Perform ADC calibration.*
- __STATIC_INLINE void Chip_ADC_SelectTempSensorInput (LPC_ADC_T ∗pADC)

    *Selects Temperature sensor as the input for Channel 0.*
- __STATIC_INLINE void Chip_ADC_SetSequencerBits (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex, uint32_t bits)

    *Helper function for safely setting ADC sequencer register bits.*
- __STATIC_INLINE void Chip_ADC_ClearSequencerBits (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex, uint32_t bits)

    *Helper function for safely clearing ADC sequencer register bits.*
- __STATIC_INLINE void Chip_ADC_SetupSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex, uint32_t options)

    *Sets up ADC conversion sequencer A or B.*
- __STATIC_INLINE uint32_t Chip_ADC_GetSequencerCtrl (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

    *Get sequenceX control register value.*
- __STATIC_INLINE void Chip_ADC_EnableSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

    *Enables a sequencer.*
- __STATIC_INLINE void Chip_ADC_DisableSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

    *Disables a sequencer.*
- __STATIC_INLINE void Chip_ADC_StartSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

    *Forces a sequencer trigger event (software trigger of ADC)*
- __STATIC_INLINE void Chip_ADC_StartBurstSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

    *Starts sequencer burst mode.*
- __STATIC_INLINE void Chip_ADC_StopBurstSequencer (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

    *Stops sequencer burst mode.*
- __STATIC_INLINE uint32_t Chip_ADC_GetGlobalDataReg (LPC_ADC_T ∗pADC, ADC_SEQ_IDX_T seqIndex)

    *Read a ADC sequence global data register.*
- __STATIC_INLINE uint32_t Chip_ADC_GetDataReg (LPC_ADC_T ∗pADC, uint8_t index)

    *Read a ADC data register.*

- __STATIC_INLINE void Chip_ADC_SetThrLowValue (LPC_ADC_T *pADC, uint8_t thrnum, uint16_t value)

  *Set Threshold low value in ADC.*

- __STATIC_INLINE void Chip_ADC_SetThrHighValue (LPC_ADC_T *pADC, uint8_t thrnum, uint16_t value)

  *Set Threshold high value in ADC.*

- __STATIC_INLINE void Chip_ADC_SelectTH0Channels (LPC_ADC_T *pADC, uint32_t channels)

  *Select threshold 0 values for comparison for selected channels.*

- __STATIC_INLINE void Chip_ADC_SelectTH1Channels (LPC_ADC_T *pADC, uint32_t channels)

  *Select threshold 1 value for comparison for selected channels.*

- __STATIC_INLINE void Chip_ADC_EnableInt (LPC_ADC_T *pADC, uint32_t intMask)

  *Enable interrupts in ADC (sequencers A/B and overrun)*

- __STATIC_INLINE void Chip_ADC_DisableInt (LPC_ADC_T *pADC, uint32_t intMask)

  *Disable interrupts in ADC (sequencers A/B and overrun)*

- __STATIC_INLINE void Chip_ADC_SetThresholdInt (LPC_ADC_T *pADC, uint8_t ch, ADC_INTEN_THCMP_T thInt)

  *Enable a threshold event interrupt in ADC.*

- __STATIC_INLINE uint32_t Chip_ADC_GetFlags (LPC_ADC_T *pADC)

  *Get flags register in ADC.*

- __STATIC_INLINE void Chip_ADC_ClearFlags (LPC_ADC_T *pADC, uint32_t flags)

  *Clear flags register in ADC.*

- __STATIC_INLINE void Chip_ADC_SetTHRSELBits (LPC_ADC_T *pADC, uint32_t mask)

  *Set Threshold selection bits.*

- __STATIC_INLINE void Chip_ADC_ClearTHRSELBits (LPC_ADC_T *pADC, uint32_t mask)

  *Clear Threshold selection bits.*

## 8.2 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/chip.h File Reference

```
#include "lpc_types.h"
```

```
#include "cmsis.h"
#include "lpc_assert.h"
#include "romapi_5411x.h"
#include "syscon_5411x.h"
#include "cpuctrl_5411x.h"
#include "clock_5411x.h"
#include "pmu_5411x.h"
#include "iocon_5411x.h"
#include "pinint_5411x.h"
#include "inmux_5411x.h"
#include "crc_5411x.h"
#include "gpio_5411x.h"
#include "mrt_5411x.h"
#include "wwdt_5411x.h"
#include "sct_5411x.h"
#include "sct_pwm_5411x.h"
#include "rtc_5411x.h"
#include "timer_5411x.h"
#include "utick_5411x.h"
#include "gpiogroup_5411x.h"
#include "mailbox_5411x.h"
#include "fpu_init.h"
#include "power_lib_5411x.h"
#include "flexcomm_5411x.h"
#include "usbd_5411x.h"
#include "adc_5411x.h"
#include "dma_5411x.h"
#include "dma_service_5411x.h"
#include "dmic_5411x.h"
#include "uart_5411x.h"
#include "spi_common_5411x.h"
#include "spim_5411x.h"
#include "spis_5411x.h"
#include "i2c_common_5411x.h"
#include "i2cm_5411x.h"
#include "i2cs_5411x.h"
#include "i2s_5411x.h"
```

## Macros

- #define LPC_FLASHMEM_BASE 0x00000000UL
- #define LPC_SRAMX_BASE 0x04000000UL
- #define LPC_SRAM0_BASE 0x20000000UL
- #define LPC_SRAM1_BASE 0x20010000UL
- #define LPC_SRAM2_BASE 0x20018000UL
- #define LPC_ROM_BASE 0x03000000UL
- #define LPC_SYSCON_BASE 0x40000000UL
- #define LPC_IOCON_BASE 0x40001000UL
- #define LPC_GPIO_GROUPINT0_BASE 0x40002000UL
- #define LPC_GPIO_GROUPINT1_BASE 0x40003000UL
- #define LPC_PIN_INT_BASE 0x40004000UL
- #define LPC_INMUX_BASE 0x40005000UL
- #define LPC_TIMER0_BASE 0x40008000UL
- #define LPC_TIMER1_BASE 0x40009000UL
- #define LPC_WWDT_BASE 0x4000C000UL

- #define LPC_MRT_BASE 0x4000D000UL
- #define LPC_UTICK_BASE 0x4000E000UL
- #define LPC_PMU_BASE 0x40020000UL
- #define LPC_TIMER2_BASE 0x40028000UL
- #define LPC_RTC_BASE 0x4002C000UL
- #define LPC_FMC_BASE 0x40034000UL
- #define LPC_ASYNC_SYSCON_BASE 0x40040000UL
- #define LPC_TIMER3_BASE 0x40048000UL
- #define LPC_TIMER4_BASE 0x40049000UL
- #define LPC_SPIFI_BASE 0x40080000UL
- #define LPC_DMA_BASE 0x40082000UL
- #define LPC_USB_BASE 0x40084000UL
- #define LPC_SCT_BASE 0x40085000UL
- #define LPC_FLEXCOMM0_BASE 0x40086000UL
- #define LPC_FLEXCOMM1_BASE 0x40087000UL
- #define LPC_FLEXCOMM2_BASE 0x40088000UL
- #define LPC_FLEXCOMM3_BASE 0x40089000UL
- #define LPC_FLEXCOMM4_BASE 0x4008A000UL
- #define LPC_MBOX_BASE 0x4008B000UL
- #define LPC_GPIO_PORT_BASE 0x4008C000UL
- #define LPC_DMIC_BASE 0x40090000UL
- #define LPC_CRC_BASE 0x40095000UL
- #define LPC_FLEXCOMM5_BASE 0x40096000UL
- #define LPC_FLEXCOMM6_BASE 0x40097000UL
- #define LPC_FLEXCOMM7_BASE 0x40098000UL
- #define LPC_ISPAP_BASE 0x4009C000UL
- #define LPC_ADC_BASE 0x400A0000UL
- #define LPC_GPIO ((LPC_GPIO_T ∗) LPC_GPIO_PORT_BASE)
- #define LPC_DMA ((LPC_DMA_T ∗) LPC_DMA_BASE)
- #define LPC_CRC ((LPC_CRC_T ∗) LPC_CRC_BASE)
- #define LPC_SCT ((LPC_SCT_T ∗) LPC_SCT_BASE)
- #define LPC_MBOX ((LPC_MBOX_T ∗) LPC_MBOX_BASE)
- #define LPC_ADC ((LPC_ADC_T ∗) LPC_ADC_BASE)
- #define LPC_PMU ((LPC_PMU_T ∗) LPC_PMU_BASE)
- #define LPC_DMIC ((LPC_DMIC_T ∗) LPC_DMIC_BASE)
- #define LPC_USB ((LPC_USB_T ∗) LPC_USB_BASE)
- #define LPC_SYSCON ((LPC_SYSCON_T ∗) LPC_SYSCON_BASE)
- #define LPC_TIMER2 ((LPC_TIMER_T ∗) LPC_TIMER2_BASE)
- #define LPC_TIMER3 ((LPC_TIMER_T ∗) LPC_TIMER3_BASE)
- #define LPC_TIMER4 ((LPC_TIMER_T ∗) LPC_TIMER4_BASE)
- #define LPC_GINT ((LPC_GPIOGROUPINT_T ∗) LPC_GPIO_GROUPINT0_BASE)
- #define LPC_PININT ((LPC_PIN_INT_T ∗) LPC_PIN_INT_BASE)
- #define LPC_IOCON ((LPC_IOCON_T ∗) LPC_IOCON_BASE)
- #define LPC_UTICK ((LPC_UTICK_T ∗) LPC_UTICK_BASE)
- #define LPC_WWDT ((LPC_WWDT_T ∗) LPC_WWDT_BASE)
- #define LPC_RTC ((LPC_RTC_T ∗) LPC_RTC_BASE)
- #define LPC_ASYNC_SYSCON ((LPC_ASYNC_SYSCON_T ∗) LPC_ASYNC_SYSCON_BASE)
- #define LPC_TIMER0 ((LPC_TIMER_T ∗) LPC_TIMER0_BASE)
- #define LPC_TIMER1 ((LPC_TIMER_T ∗) LPC_TIMER1_BASE)
- #define LPC_INMUX ((LPC_INMUX_T ∗) LPC_INMUX_BASE)
- #define LPC_MRT ((LPC_MRT_T ∗) LPC_MRT_BASE)

### Functions

- void SystemCoreClockUpdate (void)

  *Update system core and ASYNC syscon clock rate, should be called if the system has a clock rate change.*
- void Chip_SystemInit (void)

  *Set up and initialize hardware prior to call to main()*
- void Chip_SetupIrcClocking (uint32_t iFreq)

  *Clock and PLL initialization based on the internal oscillator.*
- void Chip_SetupExtInClocking (uint32_t iFreq)

  *Clock and PLL initialization based on the external clock input.*
- void Chip_SetupFROClocking (uint32_t iFreq)

  *Initialize the Core clock to given frequency (12, 48 or 96 MHz)*
- void Chip_USB_Init (void)

  *Initialize the USB bus.*
- __STATIC_INLINE void Chip_USB_TrimOff (int enable)

  *Turn of FRO clock trimming based on USB SOF.*

### Variables

- uint32_t SystemCoreClock

  *Current system clock rate, mainly used for peripherals in SYSCON.*

- const uint32_t ExtClockIn

  *Clock rate on the CLKIN pin This value is defined externally to the chip layer and contains the value in Hz for the CLKIN pin for the board. If this pin isn't used, this rate can be 0.*

## 8.3 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/clock_5411x.h File Reference

```
#include "pll_5411x.h"
```

### Macros

- #define SYSCON_FRO12MHZ_FREQ (12000000)
- #define SYSCON_FRO48MHZ_FREQ (48000000)
- #define SYSCON_FRO96MHZ_FREQ (96000000)
- #define SYSCON_WDTOSC_FREQ (500000)
- #define SYSCON_RTC_FREQ (32768)
- #define Chip_Clock_GetIntOscRate()  SYSCON_FRO12MHZ_FREQ

### Enumerations

- enum WDT_OSC_FREQ_T {
  WDT_FREQ_RESERVED, WDT_FREQ_400000, WDT_FREQ_600000, WDT_FREQ_750000,
  WDT_FREQ_900000, WDT_FREQ_1000000, WDT_FREQ_1200000, WDT_FREQ_1300000,
  WDT_FREQ_1400000, WDT_FREQ_1500000, WDT_FREQ_1600000, WDT_FREQ_1700000,
  WDT_FREQ_1800000, WDT_FREQ_1900000, WDT_FREQ_2000000, WDT_FREQ_2050000,
  WDT_FREQ_2100000, WDT_FREQ_2200000, WDT_FREQ_2250000, WDT_FREQ_2300000,
  WDT_FREQ_2400000, WDT_FREQ_2450000, WDT_FREQ_2500000, WDT_FREQ_2600000,
  WDT_FREQ_2650000, WDT_FREQ_2700000, WDT_FREQ_2800000, WDT_FREQ_2850000,
  WDT_FREQ_2900000, WDT_FREQ_2950000, WDT_FREQ_3000000, WDT_FREQ_3050000 }

*WDT Osc frequency value table.*

- enum CHIP_SYSCON_MAIN_A_CLKSRC_T { SYSCON_MAIN_A_CLKSRC_FRO12MHZ = 0, SYSCON_MAIN_A_CLKSRCA_CLKIN, SYSCON_MAIN_A_CLKSRCA_WDTOSC, SYSCON_MAIN_A_CLKSRCA_FROHF }
- enum CHIP_SYSCON_USBCLKSRC_T { SYSCON_USBCLKSRC_FROHF, SYSCON_USBCLKSRC_PLL, SYSCON_USBCLKSRC_DISABLED = 7 }

  *USB Clock source.*
- enum CHIP_SYSCON_MCLKSRC_T { SYSCON_MCLKSRC_FROHF, SYSCON_MCLKSRC_PLL, SYSCON_MCLKSRC_MCLKIN, SYSCON_MCLKSRC_DISABLED = 7 }

  *MCLK Clock sources.*
- enum CHIP_SYSCON_MAIN_B_CLKSRC_T { SYSCON_MAIN_B_CLKSRC_MAINCLKSELA = 0, SYSCON_MAIN_B_CLKSRC_PLL = 2, SYSCON_MAIN_B_CLKSRC_RTC }
- enum CHIP_SYSCON_CLKOUTSRC_T {
  SYSCON_CLKOUTSRC_MAINCLK = 0, SYSCON_CLKOUTSRC_CLKIN, SYSCON_CLKOUTSRC_WDTOSC, SYSCON_CLKOUTSRC_FROHF,
  SYSCON_CLKOUTSRC_PLL, SYSCON_CLKOUTSRC_FRO12MHZ, SYSCON_CLKOUTSRC_RTC, SYSCON_CLKOUTSRC_DISABLED }
- enum CHIP_SYSCON_CLOCK_T {
  SYSCON_CLOCK_ROM = 1, SYSCON_CLOCK_SRAM1 = 3, SYSCON_CLOCK_SRAM2, SYSCON_CLOCK_SRAMX,
  SYSCON_CLOCK_FLASH = 7, SYSCON_CLOCK_FMC, SYSCON_CLOCK_SPIFI = 10, SYSCON_CLOCK_INPUTMUX,
  SYSCON_CLOCK_IOCON = 13, SYSCON_CLOCK_GPIO0, SYSCON_CLOCK_GPIO1, SYSCON_CLOCK_PINT = 18,
  SYSCON_CLOCK_GINT, SYSCON_CLOCK_DMA, SYSCON_CLOCK_CRC, SYSCON_CLOCK_WWDT,
  SYSCON_CLOCK_RTC, SYSCON_CLOCK_MAILBOX = 26, SYSCON_CLOCK_ADC0, SYSCON_CLOCK_MRT = 32,
  SYSCON_CLOCK_SCT0 = 32 + 2, SYSCON_CLOCK_UTICK = 32 + 10, SYSCON_CLOCK_FLEXCOMM0, SYSCON_CLOCK_FLEXCOMM1,
  SYSCON_CLOCK_FLEXCOMM2, SYSCON_CLOCK_FLEXCOMM3, SYSCON_CLOCK_FLEXCOMM4, SYSCON_CLOCK_FLEXCOMM5,
  SYSCON_CLOCK_FLEXCOMM6, SYSCON_CLOCK_FLEXCOMM7, SYSCON_CLOCK_DMIC, SYSCON_CLOCK_TIMER2 = 32 + 22,
  SYSCON_CLOCK_USB = 32 + 25, SYSCON_CLOCK_TIMER0, SYSCON_CLOCK_TIMER1, SYSCON_CLOCK_TIMER3 = 128 + 13,
  SYSCON_CLOCK_TIMER4 }
- enum CHIP_SYSCON_FLEXCOMMCLKSELSRC_T {
  SYSCON_FLEXCOMMCLKSELSRC_FRO12MHZ = 0, SYSCON_FLEXCOMMCLKSELSRC_FROHF, SYSCON_FLEXCOMMCLKSELSRC_PLL, SYSCON_FLEXCOMMCLKSELSRC_MCLK,
  SYSCON_FLEXCOMMCLKSELSRC_FRG, SYSCON_FLEXCOMMCLKSELSRC_NONE = 7 }
- enum CHIP_SYSCON_ADCCLKSELSRC_T { SYSCON_ADCCLKSELSRC_MAINCLK = 0, SYSCON_ADCCLKSELSRC_SYSPLLOUT, SYSCON_ADCCLKSELSRC_FROHF }
- enum CHIP_ASYNC_SYSCON_SRC_T { SYSCON_ASYNC_MAINCLK = 0, SYSCON_ASYNC_FRO12MHZ }
- enum CHIP_SYSCON_MAINCLKSRC_T {
  SYSCON_MAINCLKSRC_FRO12MHZ = 0, SYSCON_MAINCLKSRC_CLKIN, SYSCON_MAINCLKSRC_WDTOSC, SYSCON_MAINCLKSRC_FROHF,
  SYSCON_MAINCLKSRC_PLLOUT = 6, SYSCON_MAINCLKSRC_RTC }
- enum CHIP_SYSCON_FRGCLKSRC_T {
  SYSCON_FRGCLKSRC_MAINCLK, SYSCON_FRGCLKSRC_PLL, SYSCON_FRGCLKSRC_FRO12MHZ, SYSCON_FRGCLKSRC_FROHF,
  SYSCON_FRGCLKSRC_NONE = 7 }

  *Fractional Divider clock sources.*

## Functions

- __STATIC_INLINE uint32_t Chip_Clock_GetExtClockInRate (void)

*Returns the external clock input rate.*
- __STATIC_INLINE uint32_t Chip_Clock_GetRTCOscRate (void)

    *Returns the RTC clock rate.*
- __STATIC_INLINE void Chip_Clock_SetWDTOSCRate (WDT_OSC_FREQ_T freq, uint32_t div)

    *Set the WDT Oscillator frequency and divider.*
- uint32_t Chip_Clock_GetWDTOSCRate (void)

    *Return estimated watchdog oscillator rate.*
- __STATIC_INLINE uint32_t Chip_Clock_GetFROHFRate (void)

    *Gets the HF-FRO Frequency rate.*
- __STATIC_INLINE void Chip_Clock_SetMain_A_ClockSource (CHIP_SYSCON_MAIN_A_CLKSRC_T src)

    *Set main A system clock source.*
- __STATIC_INLINE void Chip_Clock_SetUSBClockSource (CHIP_SYSCON_USBCLKSRC_T src, uint32_↩ t div)

    *Set USB clock source.*
- __STATIC_INLINE CHIP_SYSCON_USBCLKSRC_T Chip_Clock_GetUSBClockSource (void)

    *Gets the clock source used by USB.*
- __STATIC_INLINE uint32_t Chip_Clock_GetUSBClockDiv (void)

    *Gets the clock divider used by USB.*
- __STATIC_INLINE void Chip_Clock_SetMCLKClockSource (CHIP_SYSCON_MCLKSRC_T src, uint32_↩ t div)

    *Set the MCLK clock source.*
- __STATIC_INLINE uint32_t Chip_Clock_GetMCLKDiv (void)

    *Get MCLK clock div.*
- __STATIC_INLINE CHIP_SYSCON_MCLKSRC_T Chip_Clock_GetMCLKSource (void)

    *Get MCLK clock source.*
- __STATIC_INLINE void Chip_Clock_SetMCLKDirInput (void)

    *Set MCLK pin direction to INPUT.*
- __STATIC_INLINE void Chip_Clock_SetMCLKDirOutput (void)

    *Set MCLK pin direction to OUTPUT.*
- __STATIC_INLINE void Chip_Clock_SetMCLKDir (int dir)

    *Set MCLK pin direction to INPUT or OUTPUT.*
- __STATIC_INLINE int Chip_Clock_GetMCLKDir (void)
- __STATIC_INLINE CHIP_SYSCON_MAIN_A_CLKSRC_T Chip_Clock_GetMain_A_ClockSource (void)

    *Returns the main A clock source.*
- uint32_t Chip_Clock_GetMain_A_ClockRate (void)

    *Return main A clock rate.*
- __STATIC_INLINE void Chip_Clock_SetMain_B_ClockSource (CHIP_SYSCON_MAIN_B_CLKSRC_T src)

    *Set main B system clock source.*
- __STATIC_INLINE CHIP_SYSCON_MAIN_B_CLKSRC_T Chip_Clock_GetMain_B_ClockSource (void)

    *Returns the main B clock source.*
- uint32_t Chip_Clock_GetMain_B_ClockRate (void)

    *Return main B clock rate.*
- __STATIC_INLINE void Chip_Clock_SetCLKOUTSource (CHIP_SYSCON_CLKOUTSRC_T src, uint32_↩ t div)

    *Set CLKOUT clock source and divider.*
- __STATIC_INLINE CHIP_SYSCON_CLKOUTSRC_T Chip_Clock_GetCLKOUTSource (void)

    *Get CLKOUT clock source.*
- __STATIC_INLINE uint32_t Chip_Clock_GetCLKOUTDiv (void)

    *Get CLKOUT clock divider.*
- void Chip_Clock_EnablePeriphClock (CHIP_SYSCON_CLOCK_T clk)

    *Enable a system or peripheral clock.*

- void Chip_Clock_DisablePeriphClock (CHIP_SYSCON_CLOCK_T clk)

  *Disable a system or peripheral clock.*
- __STATIC_INLINE void Chip_Clock_SetSysTickClockDiv (uint32_t div)

  *Set system tick clock divider (external CLKIN as SYSTICK reference only)*
- __STATIC_INLINE uint32_t Chip_Clock_GetSysTickClockDiv (void)

  *Returns system tick clock divider.*
- uint32_t Chip_Clock_GetSysTickClockRate (void)

  *Returns the system tick rate as used with the system tick divider.*
- __STATIC_INLINE void Chip_Clock_SetSysClockDiv (uint32_t div)

  *Set system clock divider.*
- __STATIC_INLINE uint32_t Chip_Clock_GetSysClockDiv (void)

  *Get system clock divider.*
- __STATIC_INLINE void Chip_Clock_SetADCClockDiv (uint32_t div)

  *Set system tick clock divider.*
- __STATIC_INLINE uint32_t Chip_Clock_GetADCClockDiv (void)

  *Returns ADC clock divider.*
- __STATIC_INLINE void Chip_Clock_SetFLEXCOMMClockSource (uint32_t idx, CHIP_SYSCON_FLEXCOMMCLKSELSRC_T src)

  *Set the FLEXCOMM clock source.*
- __STATIC_INLINE CHIP_SYSCON_FLEXCOMMCLKSELSRC_T Chip_Clock_GetFLEXCOMMClockSource (uint32_t idx)

  *Returns the FLEXCOMM clock source.*
- uint32_t Chip_Clock_GetFLEXCOMMClockRate (uint32_t id)

  *Return FlexCOMM clock rate.*
- __STATIC_INLINE void Chip_Clock_SetADCClockSource (CHIP_SYSCON_ADCCLKSELSRC_T src)

  *Set the ADC clock source.*
- __STATIC_INLINE CHIP_SYSCON_ADCCLKSELSRC_T Chip_Clock_GetADCClockSource (void)

  *Returns the ADC clock source.*
- uint32_t Chip_Clock_GetADCClockRate (void)

  *Return ADC clock rate.*
- __STATIC_INLINE void Chip_Clock_EnableRTCOsc (void)

  *Enable the RTC 32KHz output.*
- __STATIC_INLINE void Chip_Clock_DisableRTCOsc (void)

  *Disable the RTC 32KHz output.*
- __STATIC_INLINE bool Chip_Clock_GetRTCOsc (void)
- __STATIC_INLINE void Chip_Clock_SetAsyncSysconClockSource (CHIP_ASYNC_SYSCON_SRC_T src)

  *Set asynchronous APB clock source.*
- __STATIC_INLINE CHIP_ASYNC_SYSCON_SRC_T Chip_Clock_GetAsyncSysconClockSource (void)

  *Get asynchronous APB clock source.*
- uint32_t Chip_Clock_GetAsyncSyscon_ClockRate (void)

  *Return asynchronous APB clock rate.*
- __STATIC_INLINE void Chip_Clock_SetMainClockSource (CHIP_SYSCON_MAINCLKSRC_T src)

  *Set main system clock source.*
- CHIP_SYSCON_MAINCLKSRC_T Chip_Clock_GetMainClockSource (void)

  *Get main system clock source.*
- uint32_t Chip_Clock_GetMainClockRate (void)

  *Return main clock rate.*
- uint32_t Chip_Clock_GetSystemClockRate (void)

  *Return system clock rate.*
- uint32_t Chip_Clock_GetFRGInClockRate (void)

  *Get the input clock frequency of FRG.*

- __STATIC_INLINE void Chip_Clock_SetFRGClockSource (CHIP_SYSCON_FRGCLKSRC_T src)

  *Set clock source used by FRG.*
- __STATIC_INLINE CHIP_SYSCON_FRGCLKSRC_T Chip_Clock_GetFRGClockSource (void)

  *Get clock source used by FRG.*
- uint32_t Chip_Clock_GetFRGClockRate (void)

  *Get Fraction Rate Generator (FRG) clock rate.*
- uint32_t Chip_Clock_SetFRGClockRate (uint32_t rate)

  *Set FRG rate to given rate.*

## 8.4 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/cmsis.h File Reference

```
#include "lpc_types.h"
```

## 8.5 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/cmsis_5411x.h File Reference

### Macros

- #define __CM4_REV 0x0001
- #define __MPU_PRESENT 1
- #define __NVIC_PRIO_BITS 3
- #define __Vendor_SysTickConfig 0
- #define __FPU_PRESENT 1
- #define TIMER0_IRQn CT32B0_IRQn

  *interrupt Alias*
- #define TIMER1_IRQn CT32B1_IRQn
- #define TIMER2_IRQn CT32B2_IRQn
- #define TIMER3_IRQn CT32B3_IRQn
- #define TIMER4_IRQn CT32B4_IRQn
- #define SCT_IRQn SCT0_IRQn
- #define ADC0_SEQA_IRQn ADC_SEQA_IRQn
- #define ADC0_SEQB_IRQn ADC_SEQB_IRQn
- #define ADC0_THCMP_IRQn ADC_THCMP_IRQn
- #define TIMER0_IRQHandler CT32B0_IRQHandler

  *Interrupt handler Alias.*
- #define TIMER1_IRQHandler CT32B1_IRQHandler
- #define TIMER2_IRQHandler CT32B2_IRQHandler
- #define TIMER3_IRQHandler CT32B3_IRQHandler
- #define TIMER4_IRQHandler CT32B4_IRQHandler
- #define SCT_IRQHandler SCT0_IRQHandler
- #define ADC0_SEQA_IRQHandler ADC_SEQA_IRQHandler
- #define ADC0_SEQB_IRQHandler ADC_SEQB_IRQHandler
- #define ADC0_THCMP_IRQHandler ADC_THCMP_IRQHandler

## Enumerations

- enum LPC5411X_IRQn_Type {
  Reset_IRQn = -15, NonMaskableInt_IRQn = -14, HardFault_IRQn = -13, MemoryManagement_IRQn = -12,
  BusFault_IRQn = -11, UsageFault_IRQn = -10, SVCall_IRQn = -5, DebugMonitor_IRQn = -4,
  PendSV_IRQn = -2, SysTick_IRQn = -1, WDTBOD_IRQn, DMA_IRQn,
  GINT0_IRQn, GINT1_IRQn, PIN_INT0_IRQn, PIN_INT1_IRQn,
  PIN_INT2_IRQn, PIN_INT3_IRQn, UTICK_IRQn, MRT_IRQn,
  CT32B0_IRQn, CT32B1_IRQn, SCT0_IRQn, CT32B3_IRQn,
  FLEXCOMM0_IRQn, FLEXCOMM1_IRQn, FLEXCOMM2_IRQn, FLEXCOMM3_IRQn,
  FLEXCOMM4_IRQn, FLEXCOMM5_IRQn, FLEXCOMM6_IRQn, FLEXCOMM7_IRQn,
  ADC_SEQA_IRQn, ADC_SEQB_IRQn, ADC_THCMP_IRQn, DMIC_IRQn,
  HWVAD_IRQn, USBACT_IRQn, USB_IRQn, RTC_IRQn,
  Reserved_IRQn, MAILBOX_IRQn, PIN_INT4_IRQn, PIN_INT5_IRQn,
  PIN_INT6_IRQn, PIN_INT7_IRQn, CT32B2_IRQn, CT32B4_IRQn,
  Reserved1_IRQn, SPIFI_IRQn }

## 8.6 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/cmsis_5411x_m0.h File Reference

```
#include "lpc_types.h"
```

## Macros

- #define __CM0PLUS_REV 0x0001
- #define __MPU_PRESENT 0
- #define __NVIC_PRIO_BITS 2
- #define __Vendor_SysTickConfig 0
- #define __VTOR_PRESENT 1
- #define TIMER0_IRQn CT32B0_IRQn

  *interrupt Alias*
- #define TIMER1_IRQn CT32B1_IRQn
- #define TIMER2_IRQn CT32B2_IRQn
- #define TIMER3_IRQn CT32B3_IRQn
- #define TIMER4_IRQn CT32B4_IRQn
- #define SCT_IRQn SCT0_IRQn
- #define ADC0_SEQA_IRQn ADC_SEQA_IRQn
- #define ADC0_SEQB_IRQn ADC_SEQB_IRQn
- #define ADC0_THCMP_IRQn ADC_THCMP_IRQn
- #define TIMER0_IRQHandler CT32B0_IRQHandler

  *Interrupt handler Alias.*
- #define TIMER1_IRQHandler CT32B1_IRQHandler
- #define TIMER2_IRQHandler CT32B2_IRQHandler
- #define TIMER3_IRQHandler CT32B3_IRQHandler
- #define TIMER4_IRQHandler CT32B4_IRQHandler
- #define SCT_IRQHandler SCT0_IRQHandler
- #define ADC0_SEQA_IRQHandler ADC_SEQA_IRQHandler
- #define ADC0_SEQB_IRQHandler ADC_SEQB_IRQHandler
- #define ADC0_THCMP_IRQHandler ADC_THCMP_IRQHandler

## Enumerations

- enum LPC5411X_M0_IRQn_Type {
  Reset_IRQn = -15, NonMaskableInt_IRQn = -14, HardFault_IRQn = -13, SVCall_IRQn = -5,
  PendSV_IRQn = -2, SysTick_IRQn = -1, WDTBOD_IRQn, DMA_IRQn,
  GINT0_IRQn, GINT1_IRQn, PIN_INT0_IRQn, PIN_INT1_IRQn,
  PIN_INT2_IRQn, PIN_INT3_IRQn, UTICK_IRQn, MRT_IRQn,
  CT32B0_IRQn, CT32B1_IRQn, SCT0_IRQn, CT32B3_IRQn,
  FLEXCOMM0_IRQn, FLEXCOMM1_IRQn, FLEXCOMM2_IRQn, FLEXCOMM3_IRQn,
  FLEXCOMM4_IRQn, FLEXCOMM5_IRQn, FLEXCOMM6_IRQn, FLEXCOMM7_IRQn,
  ADC_SEQA_IRQn, ADC_SEQB_IRQn, ADC_THCMP_IRQn, DMIC_IRQn,
  HWVAD, USBACT_IRQn, USB_IRQn, RTC_IRQn,
  Reserved_IRQn, MAILBOX_IRQn }

# 8.7 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm0plus.h File Reference

## 8.7.1 Detailed Description

CMSIS Cortex-M0+ Core Peripheral Access Layer Header File.

### Version

V3.20

### Date

25. February 2013

### Note

```
#include <stdint.h>
#include <core_cmInstr.h>
#include <core_cmFunc.h>
```

## Data Structures

- union APSR_Type

  *Union type to access the Application Program Status Register (APSR).*
- union IPSR_Type

  *Union type to access the Interrupt Program Status Register (IPSR).*
- union xPSR_Type

  *Union type to access the Special-Purpose Program Status Registers (xPSR).*
- union CONTROL_Type

  *Union type to access the Control Registers (CONTROL).*
- struct NVIC_Type

  *Structure type to access the Nested Vectored Interrupt Controller (NVIC).*
- struct SCB_Type

  *Structure type to access the System Control Block (SCB).*
- struct SysTick_Type

  *Structure type to access the System Timer (SysTick).*

## Macros

- #define __CORE_CM0PLUS_H_GENERIC
- #define SCB_CPUID_IMPLEMENTER_Pos 24
- #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
- #define SCB_CPUID_VARIANT_Pos 20
- #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)
- #define SCB_CPUID_ARCHITECTURE_Pos 16
- #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
- #define SCB_CPUID_PARTNO_Pos 4
- #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)
- #define SCB_CPUID_REVISION_Pos 0
- #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)
- #define SCB_ICSR_NMIPENDSET_Pos 31
- #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)
- #define SCB_ICSR_PENDSVSET_Pos 28
- #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)
- #define SCB_ICSR_PENDSVCLR_Pos 27
- #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)
- #define SCB_ICSR_PENDSTSET_Pos 26
- #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)
- #define SCB_ICSR_PENDSTCLR_Pos 25
- #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)
- #define SCB_ICSR_ISRPREEMPT_Pos 23
- #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)
- #define SCB_ICSR_ISRPENDING_Pos 22
- #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)
- #define SCB_ICSR_VECTPENDING_Pos 12
- #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)
- #define SCB_ICSR_VECTACTIVE_Pos 0
- #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)
- #define SCB_AIRCR_VECTKEY_Pos 16
- #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)
- #define SCB_AIRCR_VECTKEYSTAT_Pos 16
- #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)
- #define SCB_AIRCR_ENDIANESS_Pos 15
- #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)
- #define SCB_AIRCR_SYSRESETREQ_Pos 2
- #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)
- #define SCB_AIRCR_VECTCLRACTIVE_Pos 1
- #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)
- #define SCB_SCR_SEVONPEND_Pos 4
- #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)
- #define SCB_SCR_SLEEPDEEP_Pos 2
- #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)
- #define SCB_SCR_SLEEPONEXIT_Pos 1
- #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)
- #define SCB_CCR_STKALIGN_Pos 9
- #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)
- #define SCB_CCR_UNALIGN_TRP_Pos 3
- #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)
- #define SCB_SHCSR_SVCALLPENDED_Pos 15
- #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)
- #define SysTick_CTRL_COUNTFLAG_Pos 16
- #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)

- #define SysTick_CTRL_CLKSOURCE_Pos 2
- #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)
- #define SysTick_CTRL_TICKINT_Pos 1
- #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)
- #define SysTick_CTRL_ENABLE_Pos 0
- #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)
- #define SysTick_LOAD_RELOAD_Pos 0
- #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)
- #define SysTick_VAL_CURRENT_Pos 0
- #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define SysTick_CALIB_NOREF_Pos 31
- #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)
- #define SysTick_CALIB_SKEW_Pos 30
- #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)
- #define SysTick_CALIB_TENMS_Pos 0
- #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define SCS_BASE (0xE000E000UL)
- #define SysTick_BASE (SCS_BASE + 0x0010UL)
- #define NVIC_BASE (SCS_BASE + 0x0100UL)
- #define SCB_BASE (SCS_BASE + 0x0D00UL)
- #define SCB ((SCB_Type ∗) SCB_BASE )
- #define SysTick ((SysTick_Type ∗) SysTick_BASE )
- #define NVIC ((NVIC_Type ∗) NVIC_BASE )
- #define _BIT_SHIFT(IRQn) ( (((uint32_t)(IRQn) ) & 0x03) ∗ 8 )
- #define _SHP_IDX(IRQn) ( ((((uint32_t)(IRQn) & 0x0F)-8) >> 2) )
- #define _IP_IDX(IRQn) ( ((uint32_t)(IRQn) >> 2) )

- #define __CM0PLUS_CMSIS_VERSION_MAIN (0x03)
- #define __CM0PLUS_CMSIS_VERSION_SUB (0x20)
- #define __CM0PLUS_CMSIS_VERSION
- #define __CORTEX_M (0x00)
- #define __FPU_USED 0
- #define __CORE_CM0PLUS_H_DEPENDANT
- #define __I volatile const
- #define __O volatile
- #define __IO volatile

## Functions

- __STATIC_INLINE void NVIC_EnableIRQ (IRQn_Type IRQn)

  *Enable External Interrupt.*
- __STATIC_INLINE void NVIC_DisableIRQ (IRQn_Type IRQn)

  *Disable External Interrupt.*
- __STATIC_INLINE uint32_t NVIC_GetPendingIRQ (IRQn_Type IRQn)

  *Get Pending Interrupt.*
- __STATIC_INLINE void NVIC_SetPendingIRQ (IRQn_Type IRQn)

  *Set Pending Interrupt.*
- __STATIC_INLINE void NVIC_ClearPendingIRQ (IRQn_Type IRQn)

  *Clear Pending Interrupt.*
- __STATIC_INLINE void NVIC_SetPriority (IRQn_Type IRQn, uint32_t priority)

  *Set Interrupt Priority.*
- __STATIC_INLINE uint32_t NVIC_GetPriority (IRQn_Type IRQn)

  *Get Interrupt Priority.*

- __STATIC_INLINE void NVIC_SystemReset (void)

  *System Reset.*
- __STATIC_INLINE uint32_t SysTick_Config (uint32_t ticks)

  *System Tick Configuration.*

## 8.7.2 Macro Definition Documentation

### 8.7.2.1 #define __CM0PLUS_CMSIS_VERSION

**Value:**

```
((__CM0PLUS_CMSIS_VERSION_MAIN << 16) | \
                                       __CM0PLUS_CMSIS_VERSION_SUB)
```

CMSIS HAL version number

Definition at line 73 of file core_cm0plus.h.

### 8.7.2.2 #define __CM0PLUS_CMSIS_VERSION_MAIN (0x03)

+[31:16] CMSIS HAL main version

Definition at line 71 of file core_cm0plus.h.

### 8.7.2.3 #define __CM0PLUS_CMSIS_VERSION_SUB (0x20)

[15:0] CMSIS HAL sub version

Definition at line 72 of file core_cm0plus.h.

### 8.7.2.4 #define __CORE_CM0PLUS_H_DEPENDANT

Definition at line 135 of file core_cm0plus.h.

### 8.7.2.5 #define __CORE_CM0PLUS_H_GENERIC

Definition at line 47 of file core_cm0plus.h.

### 8.7.2.6 #define __CORTEX_M (0x00)

Cortex-M Core

Definition at line 76 of file core_cm0plus.h.

### 8.7.2.7 #define __FPU_USED 0

__FPU_USED indicates whether an FPU is used or not. This core does not support an FPU at all

Definition at line 103 of file core_cm0plus.h.

### 8.7.2.8 #define __I volatile const

Defines 'read only' permissions

Definition at line 176 of file core_cm0plus.h.

#### 8.7.2.9 #define __IO volatile

Defines 'read / write' permissions

Definition at line 179 of file core_cm0plus.h.

#### 8.7.2.10 #define __O volatile

Defines 'write only' permissions

Definition at line 178 of file core_cm0plus.h.

## 8.8 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4.h File Reference

### 8.8.1 Detailed Description

CMSIS Cortex-M4 Core Peripheral Access Layer Header File.

**Version**

V3.20

**Date**

25. February 2013

**Note**

```
#include <stdint.h>
#include <core_cmInstr.h>
#include <core_cmFunc.h>
#include <core_cm4_simd.h>
```

### Data Structures

- union APSR_Type

  *Union type to access the Application Program Status Register (APSR).*
- union IPSR_Type

  *Union type to access the Interrupt Program Status Register (IPSR).*
- union xPSR_Type

  *Union type to access the Special-Purpose Program Status Registers (xPSR).*
- union CONTROL_Type

  *Union type to access the Control Registers (CONTROL).*
- struct NVIC_Type

  *Structure type to access the Nested Vectored Interrupt Controller (NVIC).*
- struct SCB_Type

  *Structure type to access the System Control Block (SCB).*
- struct SCnSCB_Type

  *Structure type to access the System Control and ID Register not in the SCB.*
- struct SysTick_Type

*Structure type to access the System Timer (SysTick).*

- struct ITM_Type

  *Structure type to access the Instrumentation Trace Macrocell Register (ITM).*
- struct DWT_Type

  *Structure type to access the Data Watchpoint and Trace Register (DWT).*
- struct TPI_Type

  *Structure type to access the Trace Port Interface Register (TPI).*
- struct CoreDebug_Type

  *Structure type to access the Core Debug Register (CoreDebug).*

## Macros

- #define __CORE_CM4_H_GENERIC
- #define NVIC_STIR_INTID_Pos 0
- #define NVIC_STIR_INTID_Msk (0x1FFUL << NVIC_STIR_INTID_Pos)
- #define SCB_CPUID_IMPLEMENTER_Pos 24
- #define SCB_CPUID_IMPLEMENTER_Msk (0xFFUL << SCB_CPUID_IMPLEMENTER_Pos)
- #define SCB_CPUID_VARIANT_Pos 20
- #define SCB_CPUID_VARIANT_Msk (0xFUL << SCB_CPUID_VARIANT_Pos)
- #define SCB_CPUID_ARCHITECTURE_Pos 16
- #define SCB_CPUID_ARCHITECTURE_Msk (0xFUL << SCB_CPUID_ARCHITECTURE_Pos)
- #define SCB_CPUID_PARTNO_Pos 4
- #define SCB_CPUID_PARTNO_Msk (0xFFFUL << SCB_CPUID_PARTNO_Pos)
- #define SCB_CPUID_REVISION_Pos 0
- #define SCB_CPUID_REVISION_Msk (0xFUL << SCB_CPUID_REVISION_Pos)
- #define SCB_ICSR_NMIPENDSET_Pos 31
- #define SCB_ICSR_NMIPENDSET_Msk (1UL << SCB_ICSR_NMIPENDSET_Pos)
- #define SCB_ICSR_PENDSVSET_Pos 28
- #define SCB_ICSR_PENDSVSET_Msk (1UL << SCB_ICSR_PENDSVSET_Pos)
- #define SCB_ICSR_PENDSVCLR_Pos 27
- #define SCB_ICSR_PENDSVCLR_Msk (1UL << SCB_ICSR_PENDSVCLR_Pos)
- #define SCB_ICSR_PENDSTSET_Pos 26
- #define SCB_ICSR_PENDSTSET_Msk (1UL << SCB_ICSR_PENDSTSET_Pos)
- #define SCB_ICSR_PENDSTCLR_Pos 25
- #define SCB_ICSR_PENDSTCLR_Msk (1UL << SCB_ICSR_PENDSTCLR_Pos)
- #define SCB_ICSR_ISRPREEMPT_Pos 23
- #define SCB_ICSR_ISRPREEMPT_Msk (1UL << SCB_ICSR_ISRPREEMPT_Pos)
- #define SCB_ICSR_ISRPENDING_Pos 22
- #define SCB_ICSR_ISRPENDING_Msk (1UL << SCB_ICSR_ISRPENDING_Pos)
- #define SCB_ICSR_VECTPENDING_Pos 12
- #define SCB_ICSR_VECTPENDING_Msk (0x1FFUL << SCB_ICSR_VECTPENDING_Pos)
- #define SCB_ICSR_RETTOBASE_Pos 11
- #define SCB_ICSR_RETTOBASE_Msk (1UL << SCB_ICSR_RETTOBASE_Pos)
- #define SCB_ICSR_VECTACTIVE_Pos 0
- #define SCB_ICSR_VECTACTIVE_Msk (0x1FFUL << SCB_ICSR_VECTACTIVE_Pos)
- #define SCB_VTOR_TBLOFF_Pos 7
- #define SCB_VTOR_TBLOFF_Msk (0x1FFFFFFUL << SCB_VTOR_TBLOFF_Pos)
- #define SCB_AIRCR_VECTKEY_Pos 16
- #define SCB_AIRCR_VECTKEY_Msk (0xFFFFUL << SCB_AIRCR_VECTKEY_Pos)
- #define SCB_AIRCR_VECTKEYSTAT_Pos 16
- #define SCB_AIRCR_VECTKEYSTAT_Msk (0xFFFFUL << SCB_AIRCR_VECTKEYSTAT_Pos)
- #define SCB_AIRCR_ENDIANESS_Pos 15
- #define SCB_AIRCR_ENDIANESS_Msk (1UL << SCB_AIRCR_ENDIANESS_Pos)

- #define SCB_AIRCR_PRIGROUP_Pos 8
- #define SCB_AIRCR_PRIGROUP_Msk (7UL << SCB_AIRCR_PRIGROUP_Pos)
- #define SCB_AIRCR_SYSRESETREQ_Pos 2
- #define SCB_AIRCR_SYSRESETREQ_Msk (1UL << SCB_AIRCR_SYSRESETREQ_Pos)
- #define SCB_AIRCR_VECTCLRACTIVE_Pos 1
- #define SCB_AIRCR_VECTCLRACTIVE_Msk (1UL << SCB_AIRCR_VECTCLRACTIVE_Pos)
- #define SCB_AIRCR_VECTRESET_Pos 0
- #define SCB_AIRCR_VECTRESET_Msk (1UL << SCB_AIRCR_VECTRESET_Pos)
- #define SCB_SCR_SEVONPEND_Pos 4
- #define SCB_SCR_SEVONPEND_Msk (1UL << SCB_SCR_SEVONPEND_Pos)
- #define SCB_SCR_SLEEPDEEP_Pos 2
- #define SCB_SCR_SLEEPDEEP_Msk (1UL << SCB_SCR_SLEEPDEEP_Pos)
- #define SCB_SCR_SLEEPONEXIT_Pos 1
- #define SCB_SCR_SLEEPONEXIT_Msk (1UL << SCB_SCR_SLEEPONEXIT_Pos)
- #define SCB_CCR_STKALIGN_Pos 9
- #define SCB_CCR_STKALIGN_Msk (1UL << SCB_CCR_STKALIGN_Pos)
- #define SCB_CCR_BFHFNMIGN_Pos 8
- #define SCB_CCR_BFHFNMIGN_Msk (1UL << SCB_CCR_BFHFNMIGN_Pos)
- #define SCB_CCR_DIV_0_TRP_Pos 4
- #define SCB_CCR_DIV_0_TRP_Msk (1UL << SCB_CCR_DIV_0_TRP_Pos)
- #define SCB_CCR_UNALIGN_TRP_Pos 3
- #define SCB_CCR_UNALIGN_TRP_Msk (1UL << SCB_CCR_UNALIGN_TRP_Pos)
- #define SCB_CCR_USERSETMPEND_Pos 1
- #define SCB_CCR_USERSETMPEND_Msk (1UL << SCB_CCR_USERSETMPEND_Pos)
- #define SCB_CCR_NONBASETHRDENA_Pos 0
- #define SCB_CCR_NONBASETHRDENA_Msk (1UL << SCB_CCR_NONBASETHRDENA_Pos)
- #define SCB_SHCSR_USGFAULTENA_Pos 18
- #define SCB_SHCSR_USGFAULTENA_Msk (1UL << SCB_SHCSR_USGFAULTENA_Pos)
- #define SCB_SHCSR_BUSFAULTENA_Pos 17
- #define SCB_SHCSR_BUSFAULTENA_Msk (1UL << SCB_SHCSR_BUSFAULTENA_Pos)
- #define SCB_SHCSR_MEMFAULTENA_Pos 16
- #define SCB_SHCSR_MEMFAULTENA_Msk (1UL << SCB_SHCSR_MEMFAULTENA_Pos)
- #define SCB_SHCSR_SVCALLPENDED_Pos 15
- #define SCB_SHCSR_SVCALLPENDED_Msk (1UL << SCB_SHCSR_SVCALLPENDED_Pos)
- #define SCB_SHCSR_BUSFAULTPENDED_Pos 14
- #define SCB_SHCSR_BUSFAULTPENDED_Msk (1UL << SCB_SHCSR_BUSFAULTPENDED_Pos)
- #define SCB_SHCSR_MEMFAULTPENDED_Pos 13
- #define SCB_SHCSR_MEMFAULTPENDED_Msk (1UL << SCB_SHCSR_MEMFAULTPENDED_Pos)
- #define SCB_SHCSR_USGFAULTPENDED_Pos 12
- #define SCB_SHCSR_USGFAULTPENDED_Msk (1UL << SCB_SHCSR_USGFAULTPENDED_Pos)
- #define SCB_SHCSR_SYSTICKACT_Pos 11
- #define SCB_SHCSR_SYSTICKACT_Msk (1UL << SCB_SHCSR_SYSTICKACT_Pos)
- #define SCB_SHCSR_PENDSVACT_Pos 10
- #define SCB_SHCSR_PENDSVACT_Msk (1UL << SCB_SHCSR_PENDSVACT_Pos)
- #define SCB_SHCSR_MONITORACT_Pos 8
- #define SCB_SHCSR_MONITORACT_Msk (1UL << SCB_SHCSR_MONITORACT_Pos)
- #define SCB_SHCSR_SVCALLACT_Pos 7
- #define SCB_SHCSR_SVCALLACT_Msk (1UL << SCB_SHCSR_SVCALLACT_Pos)
- #define SCB_SHCSR_USGFAULTACT_Pos 3
- #define SCB_SHCSR_USGFAULTACT_Msk (1UL << SCB_SHCSR_USGFAULTACT_Pos)
- #define SCB_SHCSR_BUSFAULTACT_Pos 1
- #define SCB_SHCSR_BUSFAULTACT_Msk (1UL << SCB_SHCSR_BUSFAULTACT_Pos)
- #define SCB_SHCSR_MEMFAULTACT_Pos 0
- #define SCB_SHCSR_MEMFAULTACT_Msk (1UL << SCB_SHCSR_MEMFAULTACT_Pos)
- #define SCB_CFSR_USGFAULTSR_Pos 16

- #define SCB_CFSR_USGFAULTSR_Msk (0xFFFFUL << SCB_CFSR_USGFAULTSR_Pos)
- #define SCB_CFSR_BUSFAULTSR_Pos 8
- #define SCB_CFSR_BUSFAULTSR_Msk (0xFFUL << SCB_CFSR_BUSFAULTSR_Pos)
- #define SCB_CFSR_MEMFAULTSR_Pos 0
- #define SCB_CFSR_MEMFAULTSR_Msk (0xFFUL << SCB_CFSR_MEMFAULTSR_Pos)
- #define SCB_HFSR_DEBUGEVT_Pos 31
- #define SCB_HFSR_DEBUGEVT_Msk (1UL << SCB_HFSR_DEBUGEVT_Pos)
- #define SCB_HFSR_FORCED_Pos 30
- #define SCB_HFSR_FORCED_Msk (1UL << SCB_HFSR_FORCED_Pos)
- #define SCB_HFSR_VECTTBL_Pos 1
- #define SCB_HFSR_VECTTBL_Msk (1UL << SCB_HFSR_VECTTBL_Pos)
- #define SCB_DFSR_EXTERNAL_Pos 4
- #define SCB_DFSR_EXTERNAL_Msk (1UL << SCB_DFSR_EXTERNAL_Pos)
- #define SCB_DFSR_VCATCH_Pos 3
- #define SCB_DFSR_VCATCH_Msk (1UL << SCB_DFSR_VCATCH_Pos)
- #define SCB_DFSR_DWTTRAP_Pos 2
- #define SCB_DFSR_DWTTRAP_Msk (1UL << SCB_DFSR_DWTTRAP_Pos)
- #define SCB_DFSR_BKPT_Pos 1
- #define SCB_DFSR_BKPT_Msk (1UL << SCB_DFSR_BKPT_Pos)
- #define SCB_DFSR_HALTED_Pos 0
- #define SCB_DFSR_HALTED_Msk (1UL << SCB_DFSR_HALTED_Pos)
- #define SCnSCB_ICTR_INTLINESNUM_Pos 0
- #define SCnSCB_ICTR_INTLINESNUM_Msk (0xFUL << SCnSCB_ICTR_INTLINESNUM_Pos)
- #define SCnSCB_ACTLR_DISOOFP_Pos 9
- #define SCnSCB_ACTLR_DISOOFP_Msk (1UL << SCnSCB_ACTLR_DISOOFP_Pos)
- #define SCnSCB_ACTLR_DISFPCA_Pos 8
- #define SCnSCB_ACTLR_DISFPCA_Msk (1UL << SCnSCB_ACTLR_DISFPCA_Pos)
- #define SCnSCB_ACTLR_DISFOLD_Pos 2
- #define SCnSCB_ACTLR_DISFOLD_Msk (1UL << SCnSCB_ACTLR_DISFOLD_Pos)
- #define SCnSCB_ACTLR_DISDEFWBUF_Pos 1
- #define SCnSCB_ACTLR_DISDEFWBUF_Msk (1UL << SCnSCB_ACTLR_DISDEFWBUF_Pos)
- #define SCnSCB_ACTLR_DISMCYCINT_Pos 0
- #define SCnSCB_ACTLR_DISMCYCINT_Msk (1UL << SCnSCB_ACTLR_DISMCYCINT_Pos)
- #define SysTick_CTRL_COUNTFLAG_Pos 16
- #define SysTick_CTRL_COUNTFLAG_Msk (1UL << SysTick_CTRL_COUNTFLAG_Pos)
- #define SysTick_CTRL_CLKSOURCE_Pos 2
- #define SysTick_CTRL_CLKSOURCE_Msk (1UL << SysTick_CTRL_CLKSOURCE_Pos)
- #define SysTick_CTRL_TICKINT_Pos 1
- #define SysTick_CTRL_TICKINT_Msk (1UL << SysTick_CTRL_TICKINT_Pos)
- #define SysTick_CTRL_ENABLE_Pos 0
- #define SysTick_CTRL_ENABLE_Msk (1UL << SysTick_CTRL_ENABLE_Pos)
- #define SysTick_LOAD_RELOAD_Pos 0
- #define SysTick_LOAD_RELOAD_Msk (0xFFFFFFUL << SysTick_LOAD_RELOAD_Pos)
- #define SysTick_VAL_CURRENT_Pos 0
- #define SysTick_VAL_CURRENT_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define SysTick_CALIB_NOREF_Pos 31
- #define SysTick_CALIB_NOREF_Msk (1UL << SysTick_CALIB_NOREF_Pos)
- #define SysTick_CALIB_SKEW_Pos 30
- #define SysTick_CALIB_SKEW_Msk (1UL << SysTick_CALIB_SKEW_Pos)
- #define SysTick_CALIB_TENMS_Pos 0
- #define SysTick_CALIB_TENMS_Msk (0xFFFFFFUL << SysTick_VAL_CURRENT_Pos)
- #define ITM_TPR_PRIVMASK_Pos 0
- #define ITM_TPR_PRIVMASK_Msk (0xFUL << ITM_TPR_PRIVMASK_Pos)
- #define ITM_TCR_BUSY_Pos 23
- #define ITM_TCR_BUSY_Msk (1UL << ITM_TCR_BUSY_Pos)

- #define ITM_TCR_TraceBusID_Pos 16
- #define ITM_TCR_TraceBusID_Msk (0x7FUL << ITM_TCR_TraceBusID_Pos)
- #define ITM_TCR_GTSFREQ_Pos 10
- #define ITM_TCR_GTSFREQ_Msk (3UL << ITM_TCR_GTSFREQ_Pos)
- #define ITM_TCR_TSPrescale_Pos 8
- #define ITM_TCR_TSPrescale_Msk (3UL << ITM_TCR_TSPrescale_Pos)
- #define ITM_TCR_SWOENA_Pos 4
- #define ITM_TCR_SWOENA_Msk (1UL << ITM_TCR_SWOENA_Pos)
- #define ITM_TCR_DWTENA_Pos 3
- #define ITM_TCR_DWTENA_Msk (1UL << ITM_TCR_DWTENA_Pos)
- #define ITM_TCR_SYNCENA_Pos 2
- #define ITM_TCR_SYNCENA_Msk (1UL << ITM_TCR_SYNCENA_Pos)
- #define ITM_TCR_TSENA_Pos 1
- #define ITM_TCR_TSENA_Msk (1UL << ITM_TCR_TSENA_Pos)
- #define ITM_TCR_ITMENA_Pos 0
- #define ITM_TCR_ITMENA_Msk (1UL << ITM_TCR_ITMENA_Pos)
- #define ITM_IWR_ATVALIDM_Pos 0
- #define ITM_IWR_ATVALIDM_Msk (1UL << ITM_IWR_ATVALIDM_Pos)
- #define ITM_IRR_ATREADYM_Pos 0
- #define ITM_IRR_ATREADYM_Msk (1UL << ITM_IRR_ATREADYM_Pos)
- #define ITM_IMCR_INTEGRATION_Pos 0
- #define ITM_IMCR_INTEGRATION_Msk (1UL << ITM_IMCR_INTEGRATION_Pos)
- #define ITM_LSR_ByteAcc_Pos 2
- #define ITM_LSR_ByteAcc_Msk (1UL << ITM_LSR_ByteAcc_Pos)
- #define ITM_LSR_Access_Pos 1
- #define ITM_LSR_Access_Msk (1UL << ITM_LSR_Access_Pos)
- #define ITM_LSR_Present_Pos 0
- #define ITM_LSR_Present_Msk (1UL << ITM_LSR_Present_Pos)
- #define DWT_CTRL_NUMCOMP_Pos 28
- #define DWT_CTRL_NUMCOMP_Msk (0xFUL << DWT_CTRL_NUMCOMP_Pos)
- #define DWT_CTRL_NOTRCPKT_Pos 27
- #define DWT_CTRL_NOTRCPKT_Msk (0x1UL << DWT_CTRL_NOTRCPKT_Pos)
- #define DWT_CTRL_NOEXTTRIG_Pos 26
- #define DWT_CTRL_NOEXTTRIG_Msk (0x1UL << DWT_CTRL_NOEXTTRIG_Pos)
- #define DWT_CTRL_NOCYCCNT_Pos 25
- #define DWT_CTRL_NOCYCCNT_Msk (0x1UL << DWT_CTRL_NOCYCCNT_Pos)
- #define DWT_CTRL_NOPRFCNT_Pos 24
- #define DWT_CTRL_NOPRFCNT_Msk (0x1UL << DWT_CTRL_NOPRFCNT_Pos)
- #define DWT_CTRL_CYCEVTENA_Pos 22
- #define DWT_CTRL_CYCEVTENA_Msk (0x1UL << DWT_CTRL_CYCEVTENA_Pos)
- #define DWT_CTRL_FOLDEVTENA_Pos 21
- #define DWT_CTRL_FOLDEVTENA_Msk (0x1UL << DWT_CTRL_FOLDEVTENA_Pos)
- #define DWT_CTRL_LSUEVTENA_Pos 20
- #define DWT_CTRL_LSUEVTENA_Msk (0x1UL << DWT_CTRL_LSUEVTENA_Pos)
- #define DWT_CTRL_SLEEPEVTENA_Pos 19
- #define DWT_CTRL_SLEEPEVTENA_Msk (0x1UL << DWT_CTRL_SLEEPEVTENA_Pos)
- #define DWT_CTRL_EXCEVTENA_Pos 18
- #define DWT_CTRL_EXCEVTENA_Msk (0x1UL << DWT_CTRL_EXCEVTENA_Pos)
- #define DWT_CTRL_CPIEVTENA_Pos 17
- #define DWT_CTRL_CPIEVTENA_Msk (0x1UL << DWT_CTRL_CPIEVTENA_Pos)
- #define DWT_CTRL_EXCTRCENA_Pos 16
- #define DWT_CTRL_EXCTRCENA_Msk (0x1UL << DWT_CTRL_EXCTRCENA_Pos)
- #define DWT_CTRL_PCSAMPLENA_Pos 12
- #define DWT_CTRL_PCSAMPLENA_Msk (0x1UL << DWT_CTRL_PCSAMPLENA_Pos)
- #define DWT_CTRL_SYNCTAP_Pos 10

- #define DWT_CTRL_SYNCTAP_Msk (0x3UL << DWT_CTRL_SYNCTAP_Pos)
- #define DWT_CTRL_CYCTAP_Pos 9
- #define DWT_CTRL_CYCTAP_Msk (0x1UL << DWT_CTRL_CYCTAP_Pos)
- #define DWT_CTRL_POSTINIT_Pos 5
- #define DWT_CTRL_POSTINIT_Msk (0xFUL << DWT_CTRL_POSTINIT_Pos)
- #define DWT_CTRL_POSTPRESET_Pos 1
- #define DWT_CTRL_POSTPRESET_Msk (0xFUL << DWT_CTRL_POSTPRESET_Pos)
- #define DWT_CTRL_CYCCNTENA_Pos 0
- #define DWT_CTRL_CYCCNTENA_Msk (0x1UL << DWT_CTRL_CYCCNTENA_Pos)
- #define DWT_CPICNT_CPICNT_Pos 0
- #define DWT_CPICNT_CPICNT_Msk (0xFFUL << DWT_CPICNT_CPICNT_Pos)
- #define DWT_EXCCNT_EXCCNT_Pos 0
- #define DWT_EXCCNT_EXCCNT_Msk (0xFFUL << DWT_EXCCNT_EXCCNT_Pos)
- #define DWT_SLEEPCNT_SLEEPCNT_Pos 0
- #define DWT_SLEEPCNT_SLEEPCNT_Msk (0xFFUL << DWT_SLEEPCNT_SLEEPCNT_Pos)
- #define DWT_LSUCNT_LSUCNT_Pos 0
- #define DWT_LSUCNT_LSUCNT_Msk (0xFFUL << DWT_LSUCNT_LSUCNT_Pos)
- #define DWT_FOLDCNT_FOLDCNT_Pos 0
- #define DWT_FOLDCNT_FOLDCNT_Msk (0xFFUL << DWT_FOLDCNT_FOLDCNT_Pos)
- #define DWT_MASK_MASK_Pos 0
- #define DWT_MASK_MASK_Msk (0x1FUL << DWT_MASK_MASK_Pos)
- #define DWT_FUNCTION_MATCHED_Pos 24
- #define DWT_FUNCTION_MATCHED_Msk (0x1UL << DWT_FUNCTION_MATCHED_Pos)
- #define DWT_FUNCTION_DATAVADDR1_Pos 16
- #define DWT_FUNCTION_DATAVADDR1_Msk (0xFUL << DWT_FUNCTION_DATAVADDR1_Pos)
- #define DWT_FUNCTION_DATAVADDR0_Pos 12
- #define DWT_FUNCTION_DATAVADDR0_Msk (0xFUL << DWT_FUNCTION_DATAVADDR0_Pos)
- #define DWT_FUNCTION_DATAVSIZE_Pos 10
- #define DWT_FUNCTION_DATAVSIZE_Msk (0x3UL << DWT_FUNCTION_DATAVSIZE_Pos)
- #define DWT_FUNCTION_LNK1ENA_Pos 9
- #define DWT_FUNCTION_LNK1ENA_Msk (0x1UL << DWT_FUNCTION_LNK1ENA_Pos)
- #define DWT_FUNCTION_DATAVMATCH_Pos 8
- #define DWT_FUNCTION_DATAVMATCH_Msk (0x1UL << DWT_FUNCTION_DATAVMATCH_Pos)
- #define DWT_FUNCTION_CYCMATCH_Pos 7
- #define DWT_FUNCTION_CYCMATCH_Msk (0x1UL << DWT_FUNCTION_CYCMATCH_Pos)
- #define DWT_FUNCTION_EMITRANGE_Pos 5
- #define DWT_FUNCTION_EMITRANGE_Msk (0x1UL << DWT_FUNCTION_EMITRANGE_Pos)
- #define DWT_FUNCTION_FUNCTION_Pos 0
- #define DWT_FUNCTION_FUNCTION_Msk (0xFUL << DWT_FUNCTION_FUNCTION_Pos)
- #define TPI_ACPR_PRESCALER_Pos 0
- #define TPI_ACPR_PRESCALER_Msk (0x1FFFUL << TPI_ACPR_PRESCALER_Pos)
- #define TPI_SPPR_TXMODE_Pos 0
- #define TPI_SPPR_TXMODE_Msk (0x3UL << TPI_SPPR_TXMODE_Pos)
- #define TPI_FFSR_FtNonStop_Pos 3
- #define TPI_FFSR_FtNonStop_Msk (0x1UL << TPI_FFSR_FtNonStop_Pos)
- #define TPI_FFSR_TCPresent_Pos 2
- #define TPI_FFSR_TCPresent_Msk (0x1UL << TPI_FFSR_TCPresent_Pos)
- #define TPI_FFSR_FtStopped_Pos 1
- #define TPI_FFSR_FtStopped_Msk (0x1UL << TPI_FFSR_FtStopped_Pos)
- #define TPI_FFSR_FlInProg_Pos 0
- #define TPI_FFSR_FlInProg_Msk (0x1UL << TPI_FFSR_FlInProg_Pos)
- #define TPI_FFCR_TrigIn_Pos 8
- #define TPI_FFCR_TrigIn_Msk (0x1UL << TPI_FFCR_TrigIn_Pos)
- #define TPI_FFCR_EnFCont_Pos 1
- #define TPI_FFCR_EnFCont_Msk (0x1UL << TPI_FFCR_EnFCont_Pos)

- #define TPI_TRIGGER_TRIGGER_Pos 0
- #define TPI_TRIGGER_TRIGGER_Msk (0x1UL << TPI_TRIGGER_TRIGGER_Pos)
- #define TPI_FIFO0_ITM_ATVALID_Pos 29
- #define TPI_FIFO0_ITM_ATVALID_Msk (0x3UL << TPI_FIFO0_ITM_ATVALID_Pos)
- #define TPI_FIFO0_ITM_bytecount_Pos 27
- #define TPI_FIFO0_ITM_bytecount_Msk (0x3UL << TPI_FIFO0_ITM_bytecount_Pos)
- #define TPI_FIFO0_ETM_ATVALID_Pos 26
- #define TPI_FIFO0_ETM_ATVALID_Msk (0x3UL << TPI_FIFO0_ETM_ATVALID_Pos)
- #define TPI_FIFO0_ETM_bytecount_Pos 24
- #define TPI_FIFO0_ETM_bytecount_Msk (0x3UL << TPI_FIFO0_ETM_bytecount_Pos)
- #define TPI_FIFO0_ETM2_Pos 16
- #define TPI_FIFO0_ETM2_Msk (0xFFUL << TPI_FIFO0_ETM2_Pos)
- #define TPI_FIFO0_ETM1_Pos 8
- #define TPI_FIFO0_ETM1_Msk (0xFFUL << TPI_FIFO0_ETM1_Pos)
- #define TPI_FIFO0_ETM0_Pos 0
- #define TPI_FIFO0_ETM0_Msk (0xFFUL << TPI_FIFO0_ETM0_Pos)
- #define TPI_ITATBCTR2_ATREADY_Pos 0
- #define TPI_ITATBCTR2_ATREADY_Msk (0x1UL << TPI_ITATBCTR2_ATREADY_Pos)
- #define TPI_FIFO1_ITM_ATVALID_Pos 29
- #define TPI_FIFO1_ITM_ATVALID_Msk (0x3UL << TPI_FIFO1_ITM_ATVALID_Pos)
- #define TPI_FIFO1_ITM_bytecount_Pos 27
- #define TPI_FIFO1_ITM_bytecount_Msk (0x3UL << TPI_FIFO1_ITM_bytecount_Pos)
- #define TPI_FIFO1_ETM_ATVALID_Pos 26
- #define TPI_FIFO1_ETM_ATVALID_Msk (0x3UL << TPI_FIFO1_ETM_ATVALID_Pos)
- #define TPI_FIFO1_ETM_bytecount_Pos 24
- #define TPI_FIFO1_ETM_bytecount_Msk (0x3UL << TPI_FIFO1_ETM_bytecount_Pos)
- #define TPI_FIFO1_ITM2_Pos 16
- #define TPI_FIFO1_ITM2_Msk (0xFFUL << TPI_FIFO1_ITM2_Pos)
- #define TPI_FIFO1_ITM1_Pos 8
- #define TPI_FIFO1_ITM1_Msk (0xFFUL << TPI_FIFO1_ITM1_Pos)
- #define TPI_FIFO1_ITM0_Pos 0
- #define TPI_FIFO1_ITM0_Msk (0xFFUL << TPI_FIFO1_ITM0_Pos)
- #define TPI_ITATBCTR0_ATREADY_Pos 0
- #define TPI_ITATBCTR0_ATREADY_Msk (0x1UL << TPI_ITATBCTR0_ATREADY_Pos)
- #define TPI_ITCTRL_Mode_Pos 0
- #define TPI_ITCTRL_Mode_Msk (0x1UL << TPI_ITCTRL_Mode_Pos)
- #define TPI_DEVID_NRZVALID_Pos 11
- #define TPI_DEVID_NRZVALID_Msk (0x1UL << TPI_DEVID_NRZVALID_Pos)
- #define TPI_DEVID_MANCVALID_Pos 10
- #define TPI_DEVID_MANCVALID_Msk (0x1UL << TPI_DEVID_MANCVALID_Pos)
- #define TPI_DEVID_PTINVALID_Pos 9
- #define TPI_DEVID_PTINVALID_Msk (0x1UL << TPI_DEVID_PTINVALID_Pos)
- #define TPI_DEVID_MinBufSz_Pos 6
- #define TPI_DEVID_MinBufSz_Msk (0x7UL << TPI_DEVID_MinBufSz_Pos)
- #define TPI_DEVID_AsynClkIn_Pos 5
- #define TPI_DEVID_AsynClkIn_Msk (0x1UL << TPI_DEVID_AsynClkIn_Pos)
- #define TPI_DEVID_NrTraceInput_Pos 0
- #define TPI_DEVID_NrTraceInput_Msk (0x1FUL << TPI_DEVID_NrTraceInput_Pos)
- #define TPI_DEVTYPE_SubType_Pos 0
- #define TPI_DEVTYPE_SubType_Msk (0xFUL << TPI_DEVTYPE_SubType_Pos)
- #define TPI_DEVTYPE_MajorType_Pos 4
- #define TPI_DEVTYPE_MajorType_Msk (0xFUL << TPI_DEVTYPE_MajorType_Pos)
- #define CoreDebug_DHCSR_DBGKEY_Pos 16
- #define CoreDebug_DHCSR_DBGKEY_Msk (0xFFFFUL << CoreDebug_DHCSR_DBGKEY_Pos)
- #define CoreDebug_DHCSR_S_RESET_ST_Pos 25

- #define CoreDebug_DHCSR_S_RESET_ST_Msk (1UL << CoreDebug_DHCSR_S_RESET_ST_Pos)
- #define CoreDebug_DHCSR_S_RETIRE_ST_Pos 24
- #define CoreDebug_DHCSR_S_RETIRE_ST_Msk (1UL << CoreDebug_DHCSR_S_RETIRE_ST_Pos)
- #define CoreDebug_DHCSR_S_LOCKUP_Pos 19
- #define CoreDebug_DHCSR_S_LOCKUP_Msk (1UL << CoreDebug_DHCSR_S_LOCKUP_Pos)
- #define CoreDebug_DHCSR_S_SLEEP_Pos 18
- #define CoreDebug_DHCSR_S_SLEEP_Msk (1UL << CoreDebug_DHCSR_S_SLEEP_Pos)
- #define CoreDebug_DHCSR_S_HALT_Pos 17
- #define CoreDebug_DHCSR_S_HALT_Msk (1UL << CoreDebug_DHCSR_S_HALT_Pos)
- #define CoreDebug_DHCSR_S_REGRDY_Pos 16
- #define CoreDebug_DHCSR_S_REGRDY_Msk (1UL << CoreDebug_DHCSR_S_REGRDY_Pos)
- #define CoreDebug_DHCSR_C_SNAPSTALL_Pos 5
- #define CoreDebug_DHCSR_C_SNAPSTALL_Msk (1UL << CoreDebug_DHCSR_C_SNAPSTALL_Pos)
- #define CoreDebug_DHCSR_C_MASKINTS_Pos 3
- #define CoreDebug_DHCSR_C_MASKINTS_Msk (1UL << CoreDebug_DHCSR_C_MASKINTS_Pos)
- #define CoreDebug_DHCSR_C_STEP_Pos 2
- #define CoreDebug_DHCSR_C_STEP_Msk (1UL << CoreDebug_DHCSR_C_STEP_Pos)
- #define CoreDebug_DHCSR_C_HALT_Pos 1
- #define CoreDebug_DHCSR_C_HALT_Msk (1UL << CoreDebug_DHCSR_C_HALT_Pos)
- #define CoreDebug_DHCSR_C_DEBUGEN_Pos 0
- #define CoreDebug_DHCSR_C_DEBUGEN_Msk (1UL << CoreDebug_DHCSR_C_DEBUGEN_Pos)
- #define CoreDebug_DCRSR_REGWnR_Pos 16
- #define CoreDebug_DCRSR_REGWnR_Msk (1UL << CoreDebug_DCRSR_REGWnR_Pos)
- #define CoreDebug_DCRSR_REGSEL_Pos 0
- #define CoreDebug_DCRSR_REGSEL_Msk (0x1FUL << CoreDebug_DCRSR_REGSEL_Pos)
- #define CoreDebug_DEMCR_TRCENA_Pos 24
- #define CoreDebug_DEMCR_TRCENA_Msk (1UL << CoreDebug_DEMCR_TRCENA_Pos)
- #define CoreDebug_DEMCR_MON_REQ_Pos 19
- #define CoreDebug_DEMCR_MON_REQ_Msk (1UL << CoreDebug_DEMCR_MON_REQ_Pos)
- #define CoreDebug_DEMCR_MON_STEP_Pos 18
- #define CoreDebug_DEMCR_MON_STEP_Msk (1UL << CoreDebug_DEMCR_MON_STEP_Pos)
- #define CoreDebug_DEMCR_MON_PEND_Pos 17
- #define CoreDebug_DEMCR_MON_PEND_Msk (1UL << CoreDebug_DEMCR_MON_PEND_Pos)
- #define CoreDebug_DEMCR_MON_EN_Pos 16
- #define CoreDebug_DEMCR_MON_EN_Msk (1UL << CoreDebug_DEMCR_MON_EN_Pos)
- #define CoreDebug_DEMCR_VC_HARDERR_Pos 10
- #define CoreDebug_DEMCR_VC_HARDERR_Msk (1UL << CoreDebug_DEMCR_VC_HARDERR_Pos)
- #define CoreDebug_DEMCR_VC_INTERR_Pos 9
- #define CoreDebug_DEMCR_VC_INTERR_Msk (1UL << CoreDebug_DEMCR_VC_INTERR_Pos)
- #define CoreDebug_DEMCR_VC_BUSERR_Pos 8
- #define CoreDebug_DEMCR_VC_BUSERR_Msk (1UL << CoreDebug_DEMCR_VC_BUSERR_Pos)
- #define CoreDebug_DEMCR_VC_STATERR_Pos 7
- #define CoreDebug_DEMCR_VC_STATERR_Msk (1UL << CoreDebug_DEMCR_VC_STATERR_Pos)
- #define CoreDebug_DEMCR_VC_CHKERR_Pos 6
- #define CoreDebug_DEMCR_VC_CHKERR_Msk (1UL << CoreDebug_DEMCR_VC_CHKERR_Pos)
- #define CoreDebug_DEMCR_VC_NOCPERR_Pos 5
- #define CoreDebug_DEMCR_VC_NOCPERR_Msk (1UL << CoreDebug_DEMCR_VC_NOCPERR_Pos)
- #define CoreDebug_DEMCR_VC_MMERR_Pos 4
- #define CoreDebug_DEMCR_VC_MMERR_Msk (1UL << CoreDebug_DEMCR_VC_MMERR_Pos)
- #define CoreDebug_DEMCR_VC_CORERESET_Pos 0
- #define CoreDebug_DEMCR_VC_CORERESET_Msk (1UL << CoreDebug_DEMCR_VC_CORERESET_Pos)
- #define SCS_BASE (0xE000E000UL)
- #define ITM_BASE (0xE0000000UL)
- #define DWT_BASE (0xE0001000UL)

- #define TPI_BASE (0xE0040000UL)
- #define CoreDebug_BASE (0xE000EDF0UL)
- #define SysTick_BASE (SCS_BASE + 0x0010UL)
- #define NVIC_BASE (SCS_BASE + 0x0100UL)
- #define SCB_BASE (SCS_BASE + 0x0D00UL)
- #define SCnSCB ((SCnSCB_Type ∗) SCS_BASE )
- #define SCB ((SCB_Type ∗) SCB_BASE )
- #define SysTick ((SysTick_Type ∗) SysTick_BASE )
- #define NVIC ((NVIC_Type ∗) NVIC_BASE )
- #define ITM ((ITM_Type ∗) ITM_BASE )
- #define DWT ((DWT_Type ∗) DWT_BASE )
- #define TPI ((TPI_Type ∗) TPI_BASE )
- #define CoreDebug ((CoreDebug_Type ∗) CoreDebug_BASE)
- #define ITM_RXBUFFER_EMPTY 0x5AA55AA5

- #define __CM4_CMSIS_VERSION_MAIN (0x03)
- #define __CM4_CMSIS_VERSION_SUB (0x20)
- #define __CM4_CMSIS_VERSION
- #define __CORTEX_M (0x04)
- #define __CORE_CM4_H_DEPENDANT
- #define __I volatile const
- #define __O volatile
- #define __IO volatile

## Functions

- __STATIC_INLINE void NVIC_SetPriorityGrouping (uint32_t PriorityGroup)

  *Set Priority Grouping.*
- __STATIC_INLINE uint32_t NVIC_GetPriorityGrouping (void)

  *Get Priority Grouping.*
- __STATIC_INLINE void NVIC_EnableIRQ (IRQn_Type IRQn)

  *Enable External Interrupt.*
- __STATIC_INLINE void NVIC_DisableIRQ (IRQn_Type IRQn)

  *Disable External Interrupt.*
- __STATIC_INLINE uint32_t NVIC_GetPendingIRQ (IRQn_Type IRQn)

  *Get Pending Interrupt.*
- __STATIC_INLINE void NVIC_SetPendingIRQ (IRQn_Type IRQn)

  *Set Pending Interrupt.*
- __STATIC_INLINE void NVIC_ClearPendingIRQ (IRQn_Type IRQn)

  *Clear Pending Interrupt.*
- __STATIC_INLINE uint32_t NVIC_GetActive (IRQn_Type IRQn)

  *Get Active Interrupt.*
- __STATIC_INLINE void NVIC_SetPriority (IRQn_Type IRQn, uint32_t priority)

  *Set Interrupt Priority.*
- __STATIC_INLINE uint32_t NVIC_GetPriority (IRQn_Type IRQn)

  *Get Interrupt Priority.*
- __STATIC_INLINE uint32_t NVIC_EncodePriority (uint32_t PriorityGroup, uint32_t PreemptPriority, uint32_t SubPriority)

  *Encode Priority.*
- __STATIC_INLINE void NVIC_DecodePriority (uint32_t Priority, uint32_t PriorityGroup, uint32_t ∗pPreemptPriority, uint32_t ∗pSubPriority)

  *Decode Priority.*

- __STATIC_INLINE void NVIC_SystemReset (void)

    *System Reset.*
- __STATIC_INLINE uint32_t SysTick_Config (uint32_t ticks)

    *System Tick Configuration.*
- __STATIC_INLINE uint32_t ITM_SendChar (uint32_t ch)

    *ITM Send Character.*
- __STATIC_INLINE int32_t ITM_ReceiveChar (void)

    *ITM Receive Character.*
- __STATIC_INLINE int32_t ITM_CheckChar (void)

    *ITM Check Character.*

### Variables

- volatile int32_t ITM_RxBuffer

## 8.8.2 Macro Definition Documentation

### 8.8.2.1 #define __CM4_CMSIS_VERSION

**Value:**

```
((__CM4_CMSIS_VERSION_MAIN << 16) | \
                                     __CM4_CMSIS_VERSION_SUB          )
```

CMSIS HAL version number

Definition at line 73 of file core_cm4.h.

### 8.8.2.2 #define __CM4_CMSIS_VERSION_MAIN (0x03)

[31:16] CMSIS HAL main version

Definition at line 71 of file core_cm4.h.

### 8.8.2.3 #define __CM4_CMSIS_VERSION_SUB (0x20)

[15:0] CMSIS HAL sub version

Definition at line 72 of file core_cm4.h.

### 8.8.2.4 #define __CORE_CM4_H_DEPENDANT

__FPU_USED indicates whether an FPU is used or not. For this, __FPU_PRESENT has to be checked prior to making use of FPU specific registers and functions.

Definition at line 178 of file core_cm4.h.

### 8.8.2.5 #define __CORE_CM4_H_GENERIC

Definition at line 47 of file core_cm4.h.

#### 8.8.2.6 #define __CORTEX_M (0x04)

Cortex-M Core

Definition at line 76 of file core_cm4.h.

#### 8.8.2.7 #define __I volatile const

Defines 'read only' permissions

Definition at line 219 of file core_cm4.h.

#### 8.8.2.8 #define __IO volatile

Defines 'read / write' permissions

Definition at line 222 of file core_cm4.h.

#### 8.8.2.9 #define __O volatile

Defines 'write only' permissions

Definition at line 221 of file core_cm4.h.

## 8.9 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cm4_simd.h File Reference

### 8.9.1 Detailed Description

CMSIS Cortex-M4 SIMD Header File.

**Version**

V3.20

**Date**

25. February 2013

**Note**

### Macros

- #define __CORE_CM4_SIMD_H

### 8.9.2 Macro Definition Documentation

#### 8.9.2.1 #define __CORE_CM4_SIMD_H

Definition at line 43 of file core_cm4_simd.h.

## 8.10 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cmFunc.h File Reference

### 8.10.1 Detailed Description

CMSIS Cortex-M Core Function Access Header File.

**Version**

V3.20

**Date**

25. February 2013

**Note**

## 8.11 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/core_cmInstr.h File Reference

### 8.11.1 Detailed Description

CMSIS Cortex-M Core Instruction Access Header File.

**Version**

V3.20

**Date**

05. March 2013

**Note**

## 8.12 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/cpuctrl_5411x.h File Reference

### Enumerations

- enum CORESELECT_T { CORESELECT_M0PLUS = 0, CORESELECT_M4 }

### Functions

- __STATIC_INLINE bool Chip_CPU_IsM4Core (void)

  *Determine which MCU this code is running on.*
- void Chip_CPU_SelectMasterCore (CORESELECT_T master, CORESELECT_T ownerPower)

  *Select master core and system power control ownership.*
- bool Chip_CPU_IsMasterCore (void)

  *Determine if this core is a slave or master.*
- void Chip_CPU_CM0Boot (uint32_t ∗coentry, uint32_t ∗costackptr)

  *Setup M0+ boot and reset M0+ core.*
- void Chip_CPU_CM4Boot (uint32_t ∗coentry, uint32_t ∗costackptr)

  *Setup M4 boot and reset M4 core.*

## 8.13 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/crc_5411x.h File Reference

### Data Structures

- struct LPC_CRC_T

  *CRC register block structure.*

### Macros

- #define CRC_MODE_POLY_BITMASK ((0x03)) /∗∗ CRC polynomial Bit mask ∗/
- #define CRC_MODE_POLY_CCITT (0x00) /∗∗ Select CRC-CCITT polynomial ∗/
- #define CRC_MODE_POLY_CRC16 (0x01) /∗∗ Select CRC-16 polynomial ∗/
- #define CRC_MODE_POLY_CRC32 (0x02) /∗∗ Select CRC-32 polynomial ∗/
- #define CRC_MODE_WRDATA_BITMASK (0x03 << 2) /∗∗ CRC WR_Data Config Bit mask ∗/
- #define CRC_MODE_WRDATA_BIT_RVS (1 << 2) /∗∗ Select Bit order reverse for WR_DATA (per byte) ∗/
- #define CRC_MODE_WRDATA_CMPL (1 << 3) /∗∗ Select One's complement for WR_DATA ∗/
- #define CRC_MODE_SUM_BITMASK (0x03 << 4) /∗∗ CRC Sum Config Bit mask ∗/
- #define CRC_MODE_SUM_BIT_RVS (1 << 4) /∗∗ Select Bit order reverse for CRC_SUM ∗/
- #define CRC_MODE_SUM_CMPL (1 << 5) /∗∗ Select One's complement for CRC_SUM ∗/
- #define MODE_CFG_CCITT (0x00) /∗∗ Pre-defined mode word for default CCITT setup ∗/
- #define MODE_CFG_CRC16 (0x15) /∗∗ Pre-defined mode word for default CRC16 setup ∗/
- #define MODE_CFG_CRC32 (0x36) /∗∗ Pre-defined mode word for default CRC32 setup ∗/
- #define CRC_SEED_CCITT (0x0000FFFF)/∗∗ Initial seed value for CCITT mode ∗/
- #define CRC_SEED_CRC16 (0x00000000)/∗∗ Initial seed value for CRC16 mode ∗/
- #define CRC_SEED_CRC32 (0xFFFFFFFF)/∗∗ Initial seed value for CRC32 mode ∗/

### Enumerations

- enum CRC_POLY_T { CRC_POLY_CCITT = CRC_MODE_POLY_CCITT, CRC_POLY_CRC16 = CRC_MODE_POLY_CRC16, CRC_POLY_CRC32 = CRC_MODE_POLY_CRC32, CRC_POLY_LAST }

  *CRC polynomial.*

### Functions

- __STATIC_INLINE void Chip_CRC_Init (LPC_CRC_T ∗pCRC)

  *Initializes the CRC Engine.*
- __STATIC_INLINE void Chip_CRC_Deinit (LPC_CRC_T ∗pCRC)

  *Deinitializes the CRC Engine.*
- __STATIC_INLINE void Chip_CRC_SetPoly (LPC_CRC_T ∗pCRC, CRC_POLY_T poly, uint32_t flags)

  *Set the polynomial used for the CRC calculation.*
- __STATIC_INLINE void Chip_CRC_UseCRC16 (LPC_CRC_T ∗pCRC)

  *Sets up the CRC engine for CRC16 mode.*
- __STATIC_INLINE void Chip_CRC_UseCRC32 (LPC_CRC_T ∗pCRC)

  *Sets up the CRC engine for CRC32 mode.*
- __STATIC_INLINE void Chip_CRC_UseCCITT (LPC_CRC_T ∗pCRC)

  *Sets up the CRC engine for CCITT mode.*
- __STATIC_INLINE void Chip_CRC_UseDefaultConfig (LPC_CRC_T ∗pCRC, CRC_POLY_T poly)

  *Engage the CRC engine with defaults based on the polynomial to be used.*
- __STATIC_INLINE void Chip_CRC_SetMode (LPC_CRC_T ∗pCRC, uint32_t mode)

  *Set the CRC Mode bits.*
- __STATIC_INLINE uint32_t Chip_CRC_GetMode (LPC_CRC_T ∗pCRC)

*Get the CRC Mode bits.*
- __STATIC_INLINE void Chip_CRC_SetSeed (LPC_CRC_T ∗pCRC, uint32_t seed)

  *Set the seed bits used by the CRC_SUM register.*
- __STATIC_INLINE uint32_t Chip_CRC_GetSeed (LPC_CRC_T ∗pCRC)

  *Get the CRC seed value.*
- __STATIC_INLINE void Chip_CRC_Write8 (LPC_CRC_T ∗pCRC, uint8_t data)

  *Convenience function for writing 8-bit data to the CRC engine.*
- __STATIC_INLINE void Chip_CRC_Write16 (LPC_CRC_T ∗pCRC, uint16_t data)

  *Convenience function for writing 16-bit data to the CRC engine.*
- __STATIC_INLINE void Chip_CRC_Write32 (LPC_CRC_T ∗pCRC, uint32_t data)

  *Convenience function for writing 32-bit data to the CRC engine.*
- __STATIC_INLINE uint32_t Chip_CRC_Sum (LPC_CRC_T ∗pCRC)

  *Gets the CRC Sum based on the Mode and Seed as previously configured.*
- __STATIC_INLINE uint32_t Chip_CRC_CRC8 (LPC_CRC_T ∗pCRC, const uint8_t ∗data, uint32_t bytes)

  *Convenience function for computing a standard CCITT checksum from an 8-bit data block.*
- __STATIC_INLINE uint32_t Chip_CRC_CRC16 (LPC_CRC_T ∗pCRC, const uint16_t ∗data, uint32_↩
t hwords)

  *Convenience function for computing a standard CRC16 checksum from 16-bit data block.*
- __STATIC_INLINE uint32_t Chip_CRC_CRC32 (LPC_CRC_T ∗pCRC, const uint32_t ∗data, uint32_t words)

  *Convenience function for computing a standard CRC32 checksum from 32-bit data block.*

## 8.14 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_5411x.h File Reference

### Data Structures

- struct LPC_DMA_COMMON_T

  *DMA Controller shared registers structure.*
- struct LPC_DMA_CHANNEL_T

  *DMA Controller shared registers structure.*
- struct LPC_DMA_T

  *DMA Controller register block structure.*
- struct DMA_CHDESC_T

### Macros

- #define MAX_DMA_CHANNEL (20)
- #define DMA_INTSTAT_ACTIVEINT 0x2
- #define DMA_INTSTAT_ACTIVEERRINT 0x4
- #define DMA_ADDR(addr) ((uint32_t) (addr))
- #define DMA_CFG_PERIPHREQEN (1 << 0)
- #define DMA_CFG_HWTRIGEN (1 << 1)
- #define DMA_CFG_TRIGPOL_LOW (0 << 4)
- #define DMA_CFG_TRIGPOL_HIGH (1 << 4)
- #define DMA_CFG_TRIGTYPE_EDGE (0 << 5)
- #define DMA_CFG_TRIGTYPE_LEVEL (1 << 5)
- #define DMA_CFG_TRIGBURST_SNGL (0 << 6)
- #define DMA_CFG_TRIGBURST_BURST (1 << 6)
- #define DMA_CFG_BURSTPOWER_1 (0 << 8)
- #define DMA_CFG_BURSTPOWER_2 (1 << 8)
- #define DMA_CFG_BURSTPOWER_4 (2 << 8)
- #define DMA_CFG_BURSTPOWER_8 (3 << 8)

- #define DMA_CFG_BURSTPOWER_16 (4 << 8)
- #define DMA_CFG_BURSTPOWER_32 (5 << 8)
- #define DMA_CFG_BURSTPOWER_64 (6 << 8)
- #define DMA_CFG_BURSTPOWER_128 (7 << 8)
- #define DMA_CFG_BURSTPOWER_256 (8 << 8)
- #define DMA_CFG_BURSTPOWER_512 (9 << 8)
- #define DMA_CFG_BURSTPOWER_1024 (10 << 8)
- #define DMA_CFG_BURSTPOWER(n) ((n) << 8)
- #define DMA_CFG_SRCBURSTWRAP (1 << 14)
- #define DMA_CFG_DSTBURSTWRAP (1 << 15)
- #define DMA_CFG_CHPRIORITY(p) ((p) << 16)
- #define DMA_CTLSTAT_VALIDPENDING (1 << 0)
- #define DMA_CTLSTAT_TRIG (1 << 2)
- #define DMA_XFERCFG_CFGVALID (1 << 0)
- #define DMA_XFERCFG_RELOAD (1 << 1)
- #define DMA_XFERCFG_SWTRIG (1 << 2)
- #define DMA_XFERCFG_CLRTRIG (1 << 3)
- #define DMA_XFERCFG_SETINTA (1 << 4)
- #define DMA_XFERCFG_SETINTB (1 << 5)
- #define DMA_XFERCFG_WIDTH_8 (0 << 8)
- #define DMA_XFERCFG_WIDTH_16 (1 << 8)
- #define DMA_XFERCFG_WIDTH_32 (2 << 8)
- #define DMA_XFERCFG_SRCINC_0 (0 << 12)
- #define DMA_XFERCFG_SRCINC_1 (1 << 12)
- #define DMA_XFERCFG_SRCINC_2 (2 << 12)
- #define DMA_XFERCFG_SRCINC_4 (3 << 12)
- #define DMA_XFERCFG_DSTINC_0 (0 << 14)
- #define DMA_XFERCFG_DSTINC_1 (1 << 14)
- #define DMA_XFERCFG_DSTINC_2 (2 << 14)
- #define DMA_XFERCFG_DSTINC_4 (3 << 14)
- #define DMA_XFERCFG_XFERCOUNT(n) ((n - 1) << 16)

## Enumerations

- enum DMA_CHID_T {
  DMA_CH0, DMA_CH1, DMA_CH2, DMA_CH3,
  DMA_CH4, DMA_CH5, DMA_CH6, DMA_CH7,
  DMA_CH8, DMA_CH9, DMA_CH10, DMA_CH11,
  DMA_CH12, DMA_CH13, DMA_CH14, DMA_CH15,
  DMA_CH16, DMA_CH17, DMA_CH18, DMA_CH19,
  DMAREQ_FLEXCOMM0_RX = DMA_CH0, DMAREQ_FLEXCOMM0_TX, DMAREQ_FLEXCOMM1_RX, DMAREQ_FLEXCOMM1_TX,
  DMAREQ_FLEXCOMM2_RX, DMAREQ_FLEXCOMM2_TX, DMAREQ_FLEXCOMM3_RX, DMAREQ_FLEXCOMM3_TX,
  DMAREQ_FLEXCOMM4_RX, DMAREQ_FLEXCOMM4_TX, DMAREQ_FLEXCOMM5_RX, DMAREQ_FLEXCOMM5_TX,
  DMAREQ_FLEXCOMM6_RX, DMAREQ_FLEXCOMM6_TX, DMAREQ_FLEXCOMM7_RX, DMAREQ_FLEXCOMM7_TX,
  DMAREQ_DMIC0, DMAREQ_DMIC1, DMAREQ_SPIFI }

## Functions

- __STATIC_INLINE void Chip_DMA_Init (LPC_DMA_T ∗pDMA)

  *Initialize DMA controller.*
- __STATIC_INLINE void Chip_DMA_DeInit (LPC_DMA_T ∗pDMA)

  *De-Initialize DMA controller.*
- __STATIC_INLINE void Chip_DMA_Enable (LPC_DMA_T ∗pDMA)

  *Enable DMA controller.*
- __STATIC_INLINE void Chip_DMA_Disable (LPC_DMA_T ∗pDMA)

  *Disable DMA controller.*
- __STATIC_INLINE uint32_t Chip_DMA_GetIntStatus (LPC_DMA_T ∗pDMA)

  *Get pending interrupt or error interrupts.*
- __STATIC_INLINE void Chip_DMA_SetSRAMBase (LPC_DMA_T ∗pDMA, uint32_t base)

  *Set DMA controller SRAM base address.*
- __STATIC_INLINE uint32_t Chip_DMA_GetSRAMBase (LPC_DMA_T ∗pDMA)

  *Returns DMA controller SRAM base address.*
- __STATIC_INLINE void Chip_DMA_EnableChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Enables a single DMA channel.*
- __STATIC_INLINE void Chip_DMA_DisableChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Disables a single DMA channel.*
- __STATIC_INLINE uint32_t Chip_DMA_GetEnabledChannels (LPC_DMA_T ∗pDMA)

  *Returns all enabled DMA channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetActiveChannels (LPC_DMA_T ∗pDMA)

  *Returns all active DMA channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetBusyChannels (LPC_DMA_T ∗pDMA)

  *Returns all busy DMA channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetErrorIntChannels (LPC_DMA_T ∗pDMA)

  *Returns pending error interrupt status for all DMA channels.*
- __STATIC_INLINE void Chip_DMA_ClearErrorIntChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Clears a pending error interrupt status for a single DMA channel.*
- __STATIC_INLINE void Chip_DMA_EnableIntChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Enables a single DMA channel's interrupt used in common DMA interrupt.*
- __STATIC_INLINE void Chip_DMA_DisableIntChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Disables a single DMA channel's interrupt used in common DMA interrupt.*
- __STATIC_INLINE uint32_t Chip_DMA_GetEnableIntChannels (LPC_DMA_T ∗pDMA)

  *Returns all enabled interrupt channels.*
- __STATIC_INLINE uint32_t Chip_DMA_GetActiveIntAChannels (LPC_DMA_T ∗pDMA)

  *Returns active A interrupt status for all channels.*
- __STATIC_INLINE void Chip_DMA_ClearActiveIntAChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Clears active A interrupt status for a single channel.*
- __STATIC_INLINE uint32_t Chip_DMA_GetActiveIntBChannels (LPC_DMA_T ∗pDMA)

  *Returns active B interrupt status for all channels.*
- __STATIC_INLINE void Chip_DMA_ClearActiveIntBChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Clears active B interrupt status for a single channel.*
- __STATIC_INLINE void Chip_DMA_SetValidChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets the VALIDPENDING control bit for a single channel.*
- __STATIC_INLINE void Chip_DMA_SetTrigChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets the TRIG bit for a single channel.*
- __STATIC_INLINE void Chip_DMA_AbortChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Aborts a DMA operation for a single channel.*

- __STATIC_INLINE void Chip_DMA_SetupChannelConfig (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32↩_t cfg)

  *Setup a DMA channel configuration.*
- __STATIC_INLINE uint32_t Chip_DMA_GetChannelStatus (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Returns channel specific status flags.*
- __STATIC_INLINE void Chip_DMA_SetupChannelTransfer (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t cfg)

  *Setup a DMA channel transfer configuration.*
- __STATIC_INLINE void Chip_DMA_SetTranBits (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t mask)

  *Set DMA transfer register interrupt bits (safe)*
- __STATIC_INLINE void Chip_DMA_ClearTranBits (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t mask)

  *Clear DMA transfer register interrupt bits (safe)*
- __STATIC_INLINE void Chip_DMA_SetupChannelTransferSize (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, uint32_t trans)

  *Update the transfer size in an existing DMA channel transfer configuration.*
- __STATIC_INLINE void Chip_DMA_SetChannelValid (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets a DMA channel configuration as valid.*
- __STATIC_INLINE void Chip_DMA_SetChannelInValid (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Sets a DMA channel configuration as invalid.*
- __STATIC_INLINE void Chip_DMA_SWTriggerChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch)

  *Performs a software trigger of the DMA channel.*
- __STATIC_INLINE bool Chip_DMA_SetupTranChannel (LPC_DMA_T ∗pDMA, DMA_CHID_T ch, DMA_C↩HDESC_T ∗desc)

  *Sets up a DMA channel with the passed DMA transfer descriptor.*

### Variables

- DMA_CHDESC_T Chip_DMA_Table [MAX_DMA_CHANNEL]

## 8.15 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dma_service_5411x.h File Reference

### Data Structures

- struct DMA_PERIPHERAL_CONTEXT_T
- struct DMA_DUAL_DESCRIPTOR_T

### Typedefs

- typedef void(∗ DMA_CALLBACK_T) (int32_t)

### Functions

- void Chip_DMASERVICE_Init (DMA_CHDESC_T ∗base)

  *Initialize DMA service.*
- void Chip_DMASERVICE_Isr (void)

  *DMA service interrupt handler.*
- void Chip_DMASERVICE_RegisterCb (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, DMA_CALL↩BACK_T pCallback)

  *Register callback function.*

- void Chip_DMASERVICE_SingleBuffer (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, uint32_t p↩ Mem, uint32_t length)

    *Use Single buffer mechanism.*
- void Chip_DMASERVICE_DoubleBuffer (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, uint32_t p↩ Mem, uint32_t length, DMA_DUAL_DESCRIPTOR_T ∗pD)

    *Use double buffer mechanism.*

## 8.16 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/dmic_5411x.h File Reference

### Data Structures

- struct LPC_DMIC_Channel_Type
- struct LPC_DMIC_T
- struct DMIC_STATISTICS_T

    *DMIC statistics structure.*
- struct DMIC_CHANNEL_CONFIG_T

### Macros

- #define DMIC_FIFO_ENABLE_P 0
- #define DMIC_FIFO_RESETN_P 1
- #define DMIC_FIFO_INTREN_P 2
- #define DMIC_FIFO_DMAEN_P 3
- #define DMIC_FIFO_TLVL_P 16
- #define DMIC_FIFO_ENABLE (1<<DMIC_FIFO_ENABLE_P)
- #define DMIC_FIFO_RESETN (1<<DMIC_FIFO_RESETN_P)
- #define DMIC_FIFO_INTREN (1<<DMIC_FIFO_INTREN_P)
- #define DMIC_FIFO_DMAEN (1<<DMIC_FIFO_DMAEN_P )
- #define DMIC_FIFO_INT_P 0
- #define DMIC_FIFO_OVERRUN_P 1
- #define DMIC_FIFO_UNDERRUN_P 2
- #define DMIC_FIFO_INT (1<<DMIC_FIFO_INT_P )
- #define DMIC_FIFO_OVERRUN (1<<DMIC_FIFO_OVERRUN_P )
- #define DMIC_FIFO_UNDERRUN (1<<DMIC_FIFO_UNDERRUN_P)
- #define DMIC_PHY_FALL_P 0
- #define DMIC_PHY_HALF_P 1
- #define DMIC_PHY_FALL (1<<DMIC_PHY_FALL_P)
- #define DMIC_PHY_HALF (1<<DMIC_PHY_HALF_P)
- #define DMIC_DCPOLE_P 0
- #define DMIC_DCGAIN_REDUCE_P 4
- #define DMIC_SATURATE_AT16BIT_P 8

### Enumerations

- enum OP_MODE_T { DMIC_OP_POLL, DMIC_OP_INTR, DMIC_OP_DMA }
- enum STEREO_SIDE_T { DMIC_LEFT = 0, DMIC_RIGHT = 1 }
- enum PDM_DIV_T {
  DMIC_PDM_DIV1 = 0, DMIC_PDM_DIV2 = 1, DMIC_PDM_DIV3 = 2, DMIC_PDM_DIV4 = 3,
  DMIC_PDM_DIV6 = 4, DMIC_PDM_DIV8 = 5, DMIC_PDM_DIV12 = 6, DMIC_PDM_DIV16 = 7,
  DMIC_PDM_DIV24 = 8, DMIC_PDM_DIV32 = 9, DMIC_PDM_DIV48 = 10, DMIC_PDM_DIV64 = 11,
  DMIC_PDM_DIV96 = 12, DMIC_PDM_DIV128 = 13 }

- enum COMPENSATION_T { DMIC_COMP0_0 = 0, DMIC_COMP0_16 = 1, DMIC_COMP0_15 = 2, DMIC←_COMP0_13 = 3 }
- enum DC_REMOVAL_T { DMIC_DC_NOREMOVE = 0, DMIC_DC_CUT155 = 1, DMIC_DC_CUT78 = 2, DMIC_DC_CUT39 = 3 }
- enum DMIC_IO_T {
  pdm_dual = 0, pdm_stereo = 4, pdm_bypass = 3, pdm_bypass_clk0 = 1,
  pdm_bypass_clk1 = 2 }

### Functions

- void Chip_DMIC_Init (const CHIP_SYSCON_CLOCK_T clock, const CHIP_SYSCON_PERIPH_RESET_T reset)

  *Initialize DMIC interface.*
- void Chip_DMIC_CfgIO (LPC_DMIC_T ∗pDMIC, DMIC_IO_T cfg)

  *Configure DMIC io.*
- void Chip_DMIC_SetOpMode (LPC_DMIC_T ∗pDMIC, OP_MODE_T mode)

  *Set DMIC operating mode.*
- void Chip_DMIC_CfgChannel (LPC_DMIC_T ∗pDMIC, uint32_t channel, DMIC_CHANNEL_CONFIG_←T ∗channel_cfg)

  *Configure DMIC channel.*
- void Chip_DMIC_CfgChannelDc (LPC_DMIC_T ∗pDMIC, uint32_t channel, DC_REMOVAL_T dc_cut_level, uint32_t post_dc_gain_reduce, bool saturate16bit)

  *Configure DMIC channel DC removal setting.*
- void Chip_DMIC_Use2fs (LPC_DMIC_T ∗pDMIC, bool use2fs)

  *Configure Clock scaling.*
- void Chip_DMIC_EnableChannnel (LPC_DMIC_T ∗pDMIC, uint32_t channelmask)

  *Configure Clock scaling.*
- void Chip_DMIC_FifoChannel (LPC_DMIC_T ∗pDMIC, uint32_t channel, uint32_t trig_level, uint32_t enable, uint32_t resetn)

  *Configure fifo settings for DMIC channel.*
- __STATIC_INLINE uint32_t Chip_DMIC_FifoGetStatus (LPC_DMIC_T ∗pDMIC, uint32_t channel)

  *Get FIFO status.*
- __STATIC_INLINE void Chip_DMIC_FifoClearStatus (LPC_DMIC_T ∗pDMIC, uint32_t channel, uint32_←t mask)

  *Clear FIFO status.*
- __STATIC_INLINE uint32_t Chip_DMIC_FifoGetData (LPC_DMIC_T ∗pDMIC, uint32_t channel)

  *Get FIFO data.*

### Variables

- DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_interleaved_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_interleaved_context

## 8.17 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/error.h File Reference

### Macros

- #define offsetof(s, m)  (int) &(((s ∗) 0)->m)
- #define COMPILE_TIME_ASSERT(pred)

## Enumerations

- enum ErrorCode_t {
  LPC_OK =0, ERR_FAILED = -1, ERR_TIME_OUT = -2, ERR_BUSY = -3,
  ERR_ISP_BASE = 0x00000000, ERR_ISP_INVALID_COMMAND = ERR_ISP_BASE + 1, ERR_ISP_SRC_ADDR_ERROR, ERR_ISP_DST_ADDR_ERROR,
  ERR_ISP_SRC_ADDR_NOT_MAPPED, ERR_ISP_DST_ADDR_NOT_MAPPED, ERR_ISP_COUNT_ERROR, ERR_ISP_INVALID_SECTOR,
  ERR_ISP_SECTOR_NOT_BLANK, ERR_ISP_SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION, ERR_ISP_COMPARE_ERROR, ERR_ISP_BUSY,
  ERR_ISP_PARAM_ERROR, ERR_ISP_ADDR_ERROR, ERR_ISP_ADDR_NOT_MAPPED, ERR_ISP_CMD_LOCKED,
  ERR_ISP_INVALID_CODE, ERR_ISP_INVALID_BAUD_RATE, ERR_ISP_INVALID_STOP_BIT, ERR_ISP_CODE_READ_PROTECTION_ENABLED,
  ERR_ISP_INVALID_FLASH_UNIT, ERR_ISP_USER_CODE_CHECKSUM, ERR_ISP_SETTING_ACTIVE_PARTITION, ERR_ISP_IRC_NO_POWER,
  ERR_ISP_FLASH_NO_POWER, ERR_ISP_EEPROM_NO_POWER, ERR_ISP_EEPROM_NO_CLOCK, ERR_ISP_FLASH_NO_CLOCK,
  ERR_ISP_REINVOKE_ISP_CONFIG, ERR_API_BASE = 0x00010000, ERR_API_INVALID_PARAMS = ERR_API_BASE + 1, ERR_API_INVALID_PARAM1,
  ERR_API_INVALID_PARAM2, ERR_API_INVALID_PARAM3, ERR_API_MOD_INIT, ERR_SPIFI_BASE = 0x00020000,
  ERR_SPIFI_DEVICE_ERROR =ERR_SPIFI_BASE+1, ERR_SPIFI_INTERNAL_ERROR, ERR_SPIFI_TIMEOUT, ERR_SPIFI_OPERAND_ERROR,
  ERR_SPIFI_STATUS_PROBLEM, ERR_SPIFI_UNKNOWN_EXT, ERR_SPIFI_UNKNOWN_ID, ERR_SPIFI_UNKNOWN_TYPE,
  ERR_SPIFI_UNKNOWN_MFG, ERR_SPIFI_NO_DEVICE, ERR_SPIFI_ERASE_NEEDED, SEC_AES_NO_ERROR =0,
  ERR_SEC_AES_BASE = 0x00030000, ERR_SEC_AES_WRONG_CMD =ERR_SEC_AES_BASE+1, ERR_SEC_AES_NOT_SUPPORTED, ERR_SEC_AES_KEY_ALREADY_PROGRAMMED,
  ERR_SEC_AES_DMA_CHANNEL_CFG, ERR_SEC_AES_DMA_MUX_CFG, SEC_AES_DMA_BUSY, ERR_USBD_BASE = 0x00040000,
  ERR_USBD_INVALID_REQ = ERR_USBD_BASE + 1, ERR_USBD_UNHANDLED, ERR_USBD_STALL, ERR_USBD_SEND_ZLP,
  ERR_USBD_SEND_DATA, ERR_USBD_BAD_DESC, ERR_USBD_BAD_CFG_DESC, ERR_USBD_BAD_INTF_DESC,
  ERR_USBD_BAD_EP_DESC, ERR_USBD_BAD_MEM_BUF, ERR_USBD_TOO_MANY_CLASS_HDLR, ERR_CGU_BASE = 0x00050000,
  ERR_CGU_NOT_IMPL =ERR_CGU_BASE+1, ERR_CGU_INVALID_PARAM, ERR_CGU_INVALID_SLICE, ERR_CGU_OUTPUT_GEN,
  ERR_CGU_DIV_SRC, ERR_CGU_DIV_VAL, ERR_CGU_SRC, ERR_I2C_BASE = 0x00060000,
  ERR_I2C_BUSY = ERR_I2C_BASE, ERR_I2C_NAK, ERR_I2C_BUFFER_OVERFLOW, ERR_I2C_BYTE_COUNT_ERR,
  ERR_I2C_LOSS_OF_ARBRITRATION, ERR_I2C_SLAVE_NOT_ADDRESSED, ERR_I2C_LOSS_OF_ARBRITRATION_NAK_BIT, ERR_I2C_GENERAL_FAILURE,
  ERR_I2C_REGS_SET_TO_DEFAULT, ERR_I2C_TIMEOUT, ERR_I2C_BUFFER_UNDERFLOW, ERR_I2C_PARAM,
  ERR_OTP_BASE = 0x00070000, ERR_OTP_WR_ENABLE_INVALID = ERR_OTP_BASE+1, ERR_OTP_SOME_BITS_ALREADY_PROGRAMMED, ERR_OTP_ALL_DATA_OR_MASK_ZERO,
  ERR_OTP_WRITE_ACCESS_LOCKED, ERR_OTP_READ_DATA_MISMATCH, ERR_OTP_USB_ID_ENABLED, ERR_OTP_ETH_MAC_ENABLED,
  ERR_OTP_AES_KEYS_ENABLED, ERR_OTP_ILLEGAL_BANK, ERR_UART_BASE = 0x00080000, ERR_UART_RXD_BUSY = ERR_UART_BASE+1,
  ERR_UART_TXD_BUSY, ERR_UART_OVERRUN_FRAME_PARITY_NOISE, ERR_UART_UNDERRUN, ERR_UART_PARAM,
  ERR_UART_BAUDRATE, ERR_CAN_BASE = 0x00090000, ERR_CAN_BAD_MEM_BUF = ERR_CAN_BASE+1, ERR_CAN_INIT_FAIL,
  ERR_CANOPEN_INIT_FAIL, ERR_SPIFI_LITE_BASE = 0x000A0000, ERR_SPIFI_LITE_INVALID_ARG

UMENTS = ERR_SPIFI_LITE_BASE+1, ERR_SPIFI_LITE_BUSY,
ERR_SPIFI_LITE_MEMORY_MODE_ON, ERR_SPIFI_LITE_MEMORY_MODE_OFF, ERR_SPIFI_LITE_IN_DMA, ERR_SPIFI_LITE_NOT_IN_DMA,
PENDING_SPIFI_LITE, ERR_CLK_BASE = 0x000B0000, ERR_CLK_NOT_IMPL =ERR_CLK_BASE+1, ERR_CLK_INVALID_PARAM,
ERR_CLK_INVALID_SLICE, ERR_CLK_OUTPUT_GEN, ERR_CLK_DIV_SRC, ERR_CLK_DIV_VAL,
ERR_CLK_SRC, ERR_CLK_PLL_FIN_TOO_SMALL, ERR_CLK_PLL_FIN_TOO_LARGE, ERR_CLK_PLL_FOUT_TOO_SMALL,
ERR_CLK_PLL_FOUT_TOO_LARGE, ERR_CLK_PLL_NO_SOLUTION, ERR_CLK_PLL_MIN_PCT, ERR_CLK_PLL_MAX_PCT,
ERR_CLK_OSC_FREQ, ERR_CLK_CFG, ERR_CLK_TIMEOUT, ERR_CLK_BASE_OFF,
ERR_CLK_OFF_DEADLOCK, ERR_PWR_BASE = 0x000C0000, PWR_ERROR_ILLEGAL_MODE =ERR_PWR_BASE+1, PWR_ERROR_CLOCK_FREQ_TOO_HIGH,
PWR_ERROR_INVALID_STATE, PWR_ERROR_INVALID_CFG, PWR_ERROR_PVT_DETECT, ERR_DMA_BASE = 0x000D0000,
ERR_DMA_ERROR_INT =ERR_DMA_BASE+1, ERR_DMA_CHANNEL_NUMBER, ERR_DMA_CHANNEL_DISABLED, ERR_DMA_BUSY,
ERR_DMA_NOT_ALIGNMENT, ERR_DMA_PING_PONG_EN, ERR_DMA_CHANNEL_VALID_PENDING, ERR_DMA_PARAM,
ERR_DMA_QUEUE_EMPTY, ERR_DMA_GENERAL, ERR_SPI_BASE = 0x000E0000, ERR_SPI_BUSY =ERR_SPI_BASE,
ERR_SPI_RXOVERRUN, ERR_SPI_TXUNDERRUN, ERR_SPI_SELNASSERT, ERR_SPI_SELNDEASSERT,
ERR_SPI_CLKSTALL, ERR_SPI_PARAM, ERR_SPI_INVALID_LENGTH, ERR_ADC_BASE = 0x000F0000,
ERR_ADC_OVERRUN =ERR_ADC_BASE+1, ERR_ADC_INVALID_CHANNEL, ERR_ADC_INVALID_SEQUENCE, ERR_ADC_INVALID_SETUP,
ERR_ADC_PARAM, ERR_ADC_INVALID_LENGTH, ERR_ADC_NO_POWER, ERR_DM_BASE = 0x00100000,
ERR_DM_NOT_ENTERED =ERR_DM_BASE+1, ERR_DM_UNKNOWN_CMD, ERR_DM_COMM_FAIL }

## 8.17.1 Macro Definition Documentation

### 8.17.1.1 #define COMPILE_TIME_ASSERT( *pred* )

**Value:**

```
switch (0) { \
 case 0: \
 case pred:; }
```

Definition at line 268 of file error.h.

### 8.17.1.2 #define offsetof( *s, m* ) (int) &(((s ∗) 0)->m)

Definition at line 265 of file error.h.

## 8.17.2 Enumeration Type Documentation

### 8.17.2.1 enum ErrorCode_t

Error code returned by Boot ROM drivers/library functions

Error codes are a 32-bit value with :

- The 16 MSB contains the peripheral code number

- The 16 LSB contains an error code number associated to that peripheral

**Enumerator**

***LPC_OK*** **0x00000000** enum value returned on Success

***ERR_FAILED*** **0xFFFFFFFF** enum value returned on general failure

***ERR_TIME_OUT*** **0xFFFFFFFE** enum value returned on general timeout

***ERR_BUSY*** **0xFFFFFFFD** enum value returned when resource is busy

***ERR_ISP_BASE***

***ERR_ISP_INVALID_COMMAND***

***ERR_ISP_SRC_ADDR_ERROR***

***ERR_ISP_DST_ADDR_ERROR***

***ERR_ISP_SRC_ADDR_NOT_MAPPED***

***ERR_ISP_DST_ADDR_NOT_MAPPED***

***ERR_ISP_COUNT_ERROR***

***ERR_ISP_INVALID_SECTOR***

***ERR_ISP_SECTOR_NOT_BLANK***

***ERR_ISP_SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION***

***ERR_ISP_COMPARE_ERROR***

***ERR_ISP_BUSY***

***ERR_ISP_PARAM_ERROR***

***ERR_ISP_ADDR_ERROR***

***ERR_ISP_ADDR_NOT_MAPPED***

***ERR_ISP_CMD_LOCKED***

***ERR_ISP_INVALID_CODE***

***ERR_ISP_INVALID_BAUD_RATE***

***ERR_ISP_INVALID_STOP_BIT***

***ERR_ISP_CODE_READ_PROTECTION_ENABLED***

***ERR_ISP_INVALID_FLASH_UNIT***

***ERR_ISP_USER_CODE_CHECKSUM***

***ERR_ISP_SETTING_ACTIVE_PARTITION***

***ERR_ISP_IRC_NO_POWER***

***ERR_ISP_FLASH_NO_POWER***

***ERR_ISP_EEPROM_NO_POWER***

***ERR_ISP_EEPROM_NO_CLOCK***

***ERR_ISP_FLASH_NO_CLOCK***

***ERR_ISP_REINVOKE_ISP_CONFIG***

***ERR_API_BASE***

***ERR_API_INVALID_PARAMS*** **0x00010001** Invalid parameters

***ERR_API_INVALID_PARAM1*** **0x00010002** PARAM1 is invalid

***ERR_API_INVALID_PARAM2*** **0x00010003** PARAM2 is invalid

***ERR_API_INVALID_PARAM3*** **0x00010004** PARAM3 is invalid

***ERR_API_MOD_INIT*** **0x00010005** API is called before module init

***ERR_SPIFI_BASE***

***ERR_SPIFI_DEVICE_ERROR***

***ERR_SPIFI_INTERNAL_ERROR***

***ERR_SPIFI_TIMEOUT***

***ERR_SPIFI_OPERAND_ERROR***

***ERR_SPIFI_STATUS_PROBLEM***

***ERR_SPIFI_UNKNOWN_EXT***

***ERR_SPIFI_UNKNOWN_ID***

***ERR_SPIFI_UNKNOWN_TYPE***

***ERR_SPIFI_UNKNOWN_MFG***

***ERR_SPIFI_NO_DEVICE***

***ERR_SPIFI_ERASE_NEEDED***

***SEC_AES_NO_ERROR***

***ERR_SEC_AES_BASE***

***ERR_SEC_AES_WRONG_CMD***

***ERR_SEC_AES_NOT_SUPPORTED***

***ERR_SEC_AES_KEY_ALREADY_PROGRAMMED***

***ERR_SEC_AES_DMA_CHANNEL_CFG***

***ERR_SEC_AES_DMA_MUX_CFG***

***SEC_AES_DMA_BUSY***

***ERR_USBD_BASE***

***ERR_USBD_INVALID_REQ*** **0x00040001** invalid request

***ERR_USBD_UNHANDLED*** **0x00040002** Callback did not process the event

***ERR_USBD_STALL*** **0x00040003** Stall the endpoint on which the call back is called

***ERR_USBD_SEND_ZLP*** **0x00040004** Send ZLP packet on the endpoint on which the call back is called

***ERR_USBD_SEND_DATA*** **0x00040005** Send data packet on the endpoint on which the call back is called

***ERR_USBD_BAD_DESC*** **0x00040006** Bad descriptor

***ERR_USBD_BAD_CFG_DESC*** **0x00040007** Bad config descriptor

***ERR_USBD_BAD_INTF_DESC*** **0x00040008** Bad interface descriptor

***ERR_USBD_BAD_EP_DESC*** **0x00040009** Bad endpoint descriptor

***ERR_USBD_BAD_MEM_BUF*** **0x0004000a** Bad alignment of buffer passed.

***ERR_USBD_TOO_MANY_CLASS_HDLR*** **0x0004000b** Too many class handlers.

***ERR_CGU_BASE***

***ERR_CGU_NOT_IMPL***

***ERR_CGU_INVALID_PARAM***

***ERR_CGU_INVALID_SLICE***

***ERR_CGU_OUTPUT_GEN***

***ERR_CGU_DIV_SRC***

***ERR_CGU_DIV_VAL***

***ERR_CGU_SRC***

***ERR_I2C_BASE***

***ERR_I2C_BUSY***

***ERR_I2C_NAK***

***ERR_I2C_BUFFER_OVERFLOW***

***ERR_I2C_BYTE_COUNT_ERR***

***ERR_I2C_LOSS_OF_ARBRITRATION***

***ERR_I2C_SLAVE_NOT_ADDRESSED***

***ERR_I2C_LOSS_OF_ARBRITRATION_NAK_BIT***

***ERR_I2C_GENERAL_FAILURE***

***ERR_I2C_REGS_SET_TO_DEFAULT***

***ERR_I2C_TIMEOUT***

***ERR_I2C_BUFFER_UNDERFLOW***

***ERR_I2C_PARAM***

***ERR_OTP_BASE***

***ERR_OTP_WR_ENABLE_INVALID***

***ERR_OTP_SOME_BITS_ALREADY_PROGRAMMED***

***ERR_OTP_ALL_DATA_OR_MASK_ZERO***

***ERR_OTP_WRITE_ACCESS_LOCKED***

***ERR_OTP_READ_DATA_MISMATCH***

***ERR_OTP_USB_ID_ENABLED***

***ERR_OTP_ETH_MAC_ENABLED***

***ERR_OTP_AES_KEYS_ENABLED***

***ERR_OTP_ILLEGAL_BANK***

***ERR_UART_BASE***

***ERR_UART_RXD_BUSY***

***ERR_UART_TXD_BUSY***

***ERR_UART_OVERRUN_FRAME_PARITY_NOISE***

***ERR_UART_UNDERRUN***

***ERR_UART_PARAM***

***ERR_UART_BAUDRATE***

***ERR_CAN_BASE***

***ERR_CAN_BAD_MEM_BUF***

***ERR_CAN_INIT_FAIL***

***ERR_CANOPEN_INIT_FAIL***

***ERR_SPIFI_LITE_BASE***

***ERR_SPIFI_LITE_INVALID_ARGUMENTS***

***ERR_SPIFI_LITE_BUSY***

***ERR_SPIFI_LITE_MEMORY_MODE_ON***

***ERR_SPIFI_LITE_MEMORY_MODE_OFF***

***ERR_SPIFI_LITE_IN_DMA***

***ERR_SPIFI_LITE_NOT_IN_DMA***

***PENDING_SPIFI_LITE***

***ERR_CLK_BASE***

***ERR_CLK_NOT_IMPL***

***ERR_CLK_INVALID_PARAM***

***ERR_CLK_INVALID_SLICE***

***ERR_CLK_OUTPUT_GEN***

***ERR_CLK_DIV_SRC***

***ERR_CLK_DIV_VAL***

***ERR_CLK_SRC***

***ERR_CLK_PLL_FIN_TOO_SMALL***

***ERR_CLK_PLL_FIN_TOO_LARGE***

***ERR_CLK_PLL_FOUT_TOO_SMALL***

***ERR_CLK_PLL_FOUT_TOO_LARGE***
***ERR_CLK_PLL_NO_SOLUTION***
***ERR_CLK_PLL_MIN_PCT***
***ERR_CLK_PLL_MAX_PCT***
***ERR_CLK_OSC_FREQ***
***ERR_CLK_CFG***
***ERR_CLK_TIMEOUT***
***ERR_CLK_BASE_OFF***
***ERR_CLK_OFF_DEADLOCK***
***ERR_PWR_BASE***
***PWR_ERROR_ILLEGAL_MODE***
***PWR_ERROR_CLOCK_FREQ_TOO_HIGH***
***PWR_ERROR_INVALID_STATE***
***PWR_ERROR_INVALID_CFG***
***PWR_ERROR_PVT_DETECT***
***ERR_DMA_BASE***
***ERR_DMA_ERROR_INT***
***ERR_DMA_CHANNEL_NUMBER***
***ERR_DMA_CHANNEL_DISABLED***
***ERR_DMA_BUSY***
***ERR_DMA_NOT_ALIGNMENT***
***ERR_DMA_PING_PONG_EN***
***ERR_DMA_CHANNEL_VALID_PENDING***
***ERR_DMA_PARAM***
***ERR_DMA_QUEUE_EMPTY***
***ERR_DMA_GENERAL***
***ERR_SPI_BASE***
***ERR_SPI_BUSY***
***ERR_SPI_RXOVERRUN***
***ERR_SPI_TXUNDERRUN***
***ERR_SPI_SELNASSERT***
***ERR_SPI_SELNDEASSERT***
***ERR_SPI_CLKSTALL***
***ERR_SPI_PARAM***
***ERR_SPI_INVALID_LENGTH***
***ERR_ADC_BASE***
***ERR_ADC_OVERRUN***
***ERR_ADC_INVALID_CHANNEL***
***ERR_ADC_INVALID_SEQUENCE***
***ERR_ADC_INVALID_SETUP***
***ERR_ADC_PARAM***
***ERR_ADC_INVALID_LENGTH***
***ERR_ADC_NO_POWER***
***ERR_DM_BASE***
***ERR_DM_NOT_ENTERED***
***ERR_DM_UNKNOWN_CMD***
***ERR_DM_COMM_FAIL***

Definition at line 46 of file error.h.

## 8.18 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/flexcomm_5411x.h File Reference

```
#include "cmsis.h"
```

### Macros

- #define FLEXCOMM_LOCK (1 << 3)

  *FlexCOMM PSEL register bits.*
- #define FLEXCOMM_ID_USART (1 << 4)
- #define FLEXCOMM_ID_SPI (1 << 5)
- #define FLEXCOMM_ID_I2C (1 << 6)
- #define FLEXCOMM_ID_I2S (1 << 7)
- #define ERR_FLEXCOMM_FUNCNOTSUPPORTED -1

  *Flexcomm error and offset values.*
- #define ERR_FLEXCOMM_NOTFREE -2
- #define ERR_FLEXCOMM_INVALIDBASE -3
- #define FLEXCOMM_PSEL_OFFSET 0xFF8

### Typedefs

- typedef void LPC_FLEXCOMM_T

### Enumerations

- enum FLEXCOMM_PERIPH_T {
  FLEXCOMM_PERIPH_NONE, FLEXCOMM_PERIPH_USART, FLEXCOMM_PERIPH_SPI, FLEXCOMM_PERIPH_I2C,
  FLEXCOMM_PERIPH_I2S_TX, FLEXCOMM_PERIPH_I2S_RX }

  *FLEXCOMM Peripheral functions.*

### Functions

- __STATIC_INLINE FLEXCOMM_PERIPH_T Chip_FLEXCOMM_GetFunc (LPC_FLEXCOMM_T ∗pFCOMM)

  *Get currently enabled FLEXCOMM function.*
- __STATIC_INLINE int Chip_FLEXCOMM_IsLocked (LPC_FLEXCOMM_T ∗pFCOMM)

  *Checks if given FLEXCOMM is locked to a function.*
- __STATIC_INLINE void Chip_FLEXCOMM_Lock (LPC_FLEXCOMM_T ∗pFCOMM)

  *Lock FLEXCOMM to a function.*
- int Chip_FLEXCOMM_SetPeriph (LPC_FLEXCOMM_T ∗pFCOMM, FLEXCOMM_PERIPH_T periph, int lock)

  *Set FLEXCOMM to a peripheral function.*
- int Chip_FLEXCOMM_GetIndex (LPC_FLEXCOMM_T ∗pFCOMM)

  *Get index of the FLEXCOMM corresponding to the given base address.*
- int Chip_FLEXCOMM_Init (LPC_FLEXCOMM_T ∗pFCOMM, FLEXCOMM_PERIPH_T periph)

  *Initialize FlexCOMM and associate it with a given peripheral.*
- void Chip_FLEXCOMM_DeInit (LPC_FLEXCOMM_T ∗pFCOMM)

  *Uninitialize the FlexCOMM.*

## 8.19 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/fpu_init.h File Reference

### Functions

- void fpuInit (void)

  *Early initialization of the FPU.*

## 8.20 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/gpio_5411x.h File Reference

### Data Structures

- struct LPC_GPIO_T

  *GPIO port register block structure.*

### Functions

- __STATIC_INLINE void Chip_GPIO_Init (LPC_GPIO_T ∗pGPIO)

  *Initialize GPIO block.*
- __STATIC_INLINE void Chip_GPIO_DeInit (LPC_GPIO_T ∗pGPIO)

  *De-Initialize GPIO block.*
- __STATIC_INLINE void Chip_GPIO_WritePortBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t pin, bool setting)

  *Set a GPIO port/pin state.*
- __STATIC_INLINE void Chip_GPIO_SetPinState (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin, bool setting)

  *Set a GPIO pin state via the GPIO byte register.*
- __STATIC_INLINE bool Chip_GPIO_ReadPortBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t pin)

  *Read a GPIO pin state via the GPIO byte register.*
- __STATIC_INLINE bool Chip_GPIO_GetPinState (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Get a GPIO pin state via the GPIO byte register.*
- __STATIC_INLINE void Chip_GPIO_WriteDirBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t pin, bool setting)

  *Set GPIO direction for a single GPIO pin.*
- __STATIC_INLINE void Chip_GPIO_SetPinDIROutput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set GPIO direction for a single GPIO pin to an output.*
- __STATIC_INLINE void Chip_GPIO_SetPinDIRInput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set GPIO direction for a single GPIO pin to an input.*
- __STATIC_INLINE void Chip_GPIO_SetPinDIR (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin, bool output)

  *Set GPIO direction for a single GPIO pin.*
- __STATIC_INLINE bool Chip_GPIO_ReadDirBit (LPC_GPIO_T ∗pGPIO, uint32_t port, uint8_t bit)

  *Read a GPIO direction (out or in)*
- __STATIC_INLINE bool Chip_GPIO_GetPinDIR (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Get GPIO direction for a single GPIO pin.*
- __STATIC_INLINE void Chip_GPIO_SetDir (LPC_GPIO_T ∗pGPIO, uint8_t portNum, uint32_t bitValue, uint8_t out)

  *Set Direction for a GPIO port.*
- __STATIC_INLINE void Chip_GPIO_SetPortDIROutput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pin↩Mask)

  *Set GPIO direction for a all selected GPIO pins to an output.*
- __STATIC_INLINE void Chip_GPIO_SetPortDIRInput (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pin↩Mask)

*Set GPIO direction for a all selected GPIO pins to an input.*

- __STATIC_INLINE void Chip_GPIO_SetPortDIR (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pinMask, bool outSet)

  *Set GPIO direction for a all selected GPIO pins to an input or output.*
- __STATIC_INLINE uint32_t Chip_GPIO_GetPortDIR (LPC_GPIO_T ∗pGPIO, uint8_t port)

  *Get GPIO direction for a all GPIO pins.*
- __STATIC_INLINE void Chip_GPIO_SetPortMask (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t mask)

  *Set GPIO port mask value for GPIO masked read and write.*
- __STATIC_INLINE uint32_t Chip_GPIO_GetPortMask (LPC_GPIO_T ∗pGPIO, uint8_t port)

  *Get GPIO port mask value used for GPIO masked read and write.*
- __STATIC_INLINE void Chip_GPIO_SetPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t value)

  *Set all GPIO raw pin states (regardless of masking)*
- __STATIC_INLINE uint32_t Chip_GPIO_GetPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port)

  *Get all GPIO raw pin states (regardless of masking)*
- __STATIC_INLINE void Chip_GPIO_SetMaskedPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_↩
t value)

  *Set all GPIO pin states, but mask via the MASKP0 register.*
- __STATIC_INLINE uint32_t Chip_GPIO_GetMaskedPortValue (LPC_GPIO_T ∗pGPIO, uint8_t port)

  *Get all GPIO pin statesm but mask via the MASKP0 register.*
- __STATIC_INLINE void Chip_GPIO_SetValue (LPC_GPIO_T ∗pGPIO, uint8_t portNum, uint32_t bitValue)

  *Set a GPIO port/bit to the high state.*
- __STATIC_INLINE void Chip_GPIO_SetPortOutHigh (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pins)

  *Set selected GPIO output pins to the high state.*
- __STATIC_INLINE void Chip_GPIO_SetPinOutHigh (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set an individual GPIO output pin to the high state.*
- __STATIC_INLINE void Chip_GPIO_ClearValue (LPC_GPIO_T ∗pGPIO, uint8_t portNum, uint32_t bitValue)

  *Set a GPIO port/bit to the low state.*
- __STATIC_INLINE void Chip_GPIO_SetPortOutLow (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pins)

  *Set selected GPIO output pins to the low state.*
- __STATIC_INLINE void Chip_GPIO_SetPinOutLow (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Set an individual GPIO output pin to the low state.*
- __STATIC_INLINE void Chip_GPIO_SetPortToggle (LPC_GPIO_T ∗pGPIO, uint8_t port, uint32_t pins)

  *Toggle selected GPIO output pins to the opposite state.*
- __STATIC_INLINE void Chip_GPIO_SetPinToggle (LPC_GPIO_T ∗pGPIO, uint8_t port, uint8_t pin)

  *Toggle an individual GPIO output pin to the opposite state.*
- __STATIC_INLINE uint32_t Chip_GPIO_ReadValue (LPC_GPIO_T ∗pGPIO, uint8_t portNum)

  *Read current bit states for the selected port.*

## 8.21 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/gpiogroup_5411x.h File Reference

### Data Structures

- struct LPC_GPIOGROUPINT_T

  *GPIO grouped interrupt register block structure.*

### Macros

- #define GPIOGR_INT (1 << 0)
- #define GPIOGR_COMB (1 << 1)
- #define GPIOGR_TRIG (1 << 2)

### Functions

- __STATIC_INLINE void Chip_GPIOGP_Init (LPC_GPIOGROUPINT_T ∗pGPIOGPINT)

  *Initialize GPIO group interrupt block.*
- __STATIC_INLINE void Chip_GPIOGP_DeInit (LPC_GPIOGROUPINT_T ∗pGPIOGPINT)

  *De-Initialize GPIO group interrupt block.*
- __STATIC_INLINE void Chip_GPIOGP_ClearIntStatus (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_↩ t group)

  *Clear interrupt pending status for the selected group.*
- __STATIC_INLINE bool Chip_GPIOGP_GetIntStatus (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_↩ t group)

  *Returns current GPIO group inetrrupt pending status.*
- __STATIC_INLINE void Chip_GPIOGP_SelectOrMode (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_↩ t group)

  *Selected GPIO group functionality for trigger on any pin in group (OR mode)*
- __STATIC_INLINE void Chip_GPIOGP_SelectAndMode (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8↩ _t group)

  *Selected GPIO group functionality for trigger on all matching pins in group (AND mode)*
- __STATIC_INLINE void Chip_GPIOGP_SelectEdgeMode (LPC_GPIOGROUPINT_T ∗pGPIOGPIN↩ T, uint8_t group)

  *Selected GPIO group functionality edge trigger mode.*
- __STATIC_INLINE void Chip_GPIOGP_SelectLevelMode (LPC_GPIOGROUPINT_T ∗pGPIOGPIN↩ T, uint8_t group)

  *Selected GPIO group functionality level trigger mode.*
- __STATIC_INLINE void Chip_GPIOGP_SelectLowLevel (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8_t group, uint8_t port, uint32_t pinMask)

  *Set selected pins for the group and port to low level trigger.*
- __STATIC_INLINE void Chip_GPIOGP_SelectHighLevel (LPC_GPIOGROUPINT_T ∗pGPIOGPINT, uint8↩ _t group, uint8_t port, uint32_t pinMask)

  *Set selected pins for the group and port to high level trigger.*
- __STATIC_INLINE void Chip_GPIOGP_DisableGroupPins (LPC_GPIOGROUPINT_T ∗pGPIOGPIN↩ T, uint8_t group, uint8_t port, uint32_t pinMask)

  *Disabled selected pins for the group interrupt.*
- __STATIC_INLINE void Chip_GPIOGP_EnableGroupPins (LPC_GPIOGROUPINT_T ∗pGPIOGPIN↩ T, uint8_t group, uint8_t port, uint32_t pinMask)

  *Enable selected pins for the group interrupt.*

## 8.22 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2c_common_5411x.h File Reference

### Macros

- #define I2C_CFG_MSTEN (1 << 0)
- #define I2C_CFG_SLVEN (1 << 1)
- #define I2C_CFG_MONEN (1 << 2)
- #define I2C_CFG_TIMEOUTEN (1 << 3)
- #define I2C_CFG_MONCLKSTR (1 << 4)
- #define I2C_CFG_MASK ((uint32_t) 0x1F)
- #define I2C_STAT_MSTPENDING (1 << 0)
- #define I2C_STAT_MSTSTATE (0x7 << 1)
- #define I2C_STAT_MSTRARBLOSS (1 << 4)
- #define I2C_STAT_MSTSTSTPERR (1 << 6)
- #define I2C_STAT_SLVPENDING (1 << 8)

- #define I2C_STAT_SLVSTATE (0x3 << 9)
- #define I2C_STAT_SLVNOTSTR (1 << 11)
- #define I2C_STAT_SLVIDX (0x3 << 12)
- #define I2C_STAT_SLVSEL (1 << 14)
- #define I2C_STAT_SLVDESEL (1 << 15)
- #define I2C_STAT_MONRDY (1 << 16)
- #define I2C_STAT_MONOV (1 << 17)
- #define I2C_STAT_MONACTIVE (1 << 18)
- #define I2C_STAT_MONIDLE (1 << 19)
- #define I2C_STAT_EVENTTIMEOUT (1 << 24)
- #define I2C_STAT_SCLTIMEOUT (1 << 25)
- #define I2C_STAT_MSTCODE_IDLE (0)
- #define I2C_STAT_MSTCODE_RXREADY (1)
- #define I2C_STAT_MSTCODE_TXREADY (2)
- #define I2C_STAT_MSTCODE_NACKADR (3)
- #define I2C_STAT_MSTCODE_NACKDAT (4)
- #define I2C_STAT_SLVCODE_ADDR (0)
- #define I2C_STAT_SLVCODE_RX (1)
- #define I2C_STAT_SLVCODE_TX (2)
- #define I2C_INTENSET_MSTPENDING (1 << 0)
- #define I2C_INTENSET_MSTRARBLOSS (1 << 4)
- #define I2C_INTENSET_MSTSTSTPERR (1 << 6)
- #define I2C_INTENSET_SLVPENDING (1 << 8)
- #define I2C_INTENSET_SLVNOTSTR (1 << 11)
- #define I2C_INTENSET_SLVDESEL (1 << 15)
- #define I2C_INTENSET_MONRDY (1 << 16)
- #define I2C_INTENSET_MONOV (1 << 17)
- #define I2C_INTENSET_MONIDLE (1 << 19)
- #define I2C_INTENSET_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENSET_SCLTIMEOUT (1 << 25)
- #define I2C_INTENCLR_MSTPENDING (1 << 0)
- #define I2C_INTENCLR_MSTRARBLOSS (1 << 4)
- #define I2C_INTENCLR_MSTSTSTPERR (1 << 6)
- #define I2C_INTENCLR_SLVPENDING (1 << 8)
- #define I2C_INTENCLR_SLVNOTSTR (1 << 11)
- #define I2C_INTENCLR_SLVDESEL (1 << 15)
- #define I2C_INTENCLR_MONRDY (1 << 16)
- #define I2C_INTENCLR_MONOV (1 << 17)
- #define I2C_INTENCLR_MONIDLE (1 << 19)
- #define I2C_INTENCLR_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENCLR_SCLTIMEOUT (1 << 25)
- #define I2C_TIMEOUT_VAL(n) (((uint32_t) ((n) - 1) & 0xFFF0) | 0x000F)
- #define I2C_INTSTAT_MSTPENDING (1 << 0)
- #define I2C_INTSTAT_MSTRARBLOSS (1 << 4)
- #define I2C_INTSTAT_MSTSTSTPERR (1 << 6)
- #define I2C_INTSTAT_SLVPENDING (1 << 8)
- #define I2C_INTSTAT_SLVNOTSTR (1 << 11)
- #define I2C_INTSTAT_SLVDESEL (1 << 15)
- #define I2C_INTSTAT_MONRDY (1 << 16)
- #define I2C_INTSTAT_MONOV (1 << 17)
- #define I2C_INTSTAT_MONIDLE (1 << 19)
- #define I2C_INTSTAT_EVENTTIMEOUT (1 << 24)
- #define I2C_INTSTAT_SCLTIMEOUT (1 << 25)
- #define I2C_MSTCTL_MSTCONTINUE (1 << 0)
- #define I2C_MSTCTL_MSTSTART (1 << 1)

- #define I2C_MSTCTL_MSTSTOP (1 << 2)
- #define I2C_MSTCTL_MSTDMA (1 << 3)
- #define I2C_MSTTIME_MSTSCLLOW (0x07 << 0)
- #define I2C_MSTTIME_MSTSCLHIGH (0x07 << 4)
- #define I2C_MSTDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_SLVCTL_SLVCONTINUE (1 << 0)
- #define I2C_SLVCTL_SLVNACK (1 << 1)
- #define I2C_SLVCTL_SLVDMA (1 << 3)
- #define I2C_SLVDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_SLVADR_SADISABLE (1 << 0)
- #define I2C_SLVADR_SLVADR (0x7F << 1)
- #define I2C_SLVADR_MASK ((uint32_t) 0x00FF)
- #define I2C_SLVQUAL_QUALMODE0 (1 << 0)
- #define I2C_SLVQUAL_SLVQUAL0 (0x7F << 1)
- #define I2C_MONRXDAT_DATA (0xFF << 0)
- #define I2C_MONRXDAT_MONSTART (1 << 8)
- #define I2C_MONRXDAT_MONRESTART (1 << 9)
- #define I2C_MONRXDAT_MONNACK (1 << 10)
- #define I2C_CFG_MSTEN (1 << 0)
- #define I2C_CFG_SLVEN (1 << 1)
- #define I2C_CFG_MONEN (1 << 2)
- #define I2C_CFG_TIMEOUTEN (1 << 3)
- #define I2C_CFG_MONCLKSTR (1 << 4)
- #define I2C_CFG_MASK ((uint32_t) 0x1F)
- #define I2C_STAT_MSTPENDING (1 << 0)
- #define I2C_STAT_MSTSTATE (0x7 << 1)
- #define I2C_STAT_MSTRARBLOSS (1 << 4)
- #define I2C_STAT_MSTSTSTPERR (1 << 6)
- #define I2C_STAT_SLVPENDING (1 << 8)
- #define I2C_STAT_SLVSTATE (0x3 << 9)
- #define I2C_STAT_SLVNOTSTR (1 << 11)
- #define I2C_STAT_SLVIDX (0x3 << 12)
- #define I2C_STAT_SLVSEL (1 << 14)
- #define I2C_STAT_SLVDESEL (1 << 15)
- #define I2C_STAT_MONRDY (1 << 16)
- #define I2C_STAT_MONOV (1 << 17)
- #define I2C_STAT_MONACTIVE (1 << 18)
- #define I2C_STAT_MONIDLE (1 << 19)
- #define I2C_STAT_EVENTTIMEOUT (1 << 24)
- #define I2C_STAT_SCLTIMEOUT (1 << 25)
- #define I2C_STAT_MSTCODE_IDLE (0)
- #define I2C_STAT_MSTCODE_RXREADY (1)
- #define I2C_STAT_MSTCODE_TXREADY (2)
- #define I2C_STAT_MSTCODE_NACKADR (3)
- #define I2C_STAT_MSTCODE_NACKDAT (4)
- #define I2C_STAT_SLVCODE_ADDR (0)
- #define I2C_STAT_SLVCODE_RX (1)
- #define I2C_STAT_SLVCODE_TX (2)
- #define I2C_INTENSET_MSTPENDING (1 << 0)
- #define I2C_INTENSET_MSTRARBLOSS (1 << 4)
- #define I2C_INTENSET_MSTSTSTPERR (1 << 6)
- #define I2C_INTENSET_SLVPENDING (1 << 8)
- #define I2C_INTENSET_SLVNOTSTR (1 << 11)
- #define I2C_INTENSET_SLVDESEL (1 << 15)
- #define I2C_INTENSET_MONRDY (1 << 16)

- #define I2C_INTENSET_MONOV (1 << 17)
- #define I2C_INTENSET_MONIDLE (1 << 19)
- #define I2C_INTENSET_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENSET_SCLTIMEOUT (1 << 25)
- #define I2C_INTENCLR_MSTPENDING (1 << 0)
- #define I2C_INTENCLR_MSTRARBLOSS (1 << 4)
- #define I2C_INTENCLR_MSTSTSTPERR (1 << 6)
- #define I2C_INTENCLR_SLVPENDING (1 << 8)
- #define I2C_INTENCLR_SLVNOTSTR (1 << 11)
- #define I2C_INTENCLR_SLVDESEL (1 << 15)
- #define I2C_INTENCLR_MONRDY (1 << 16)
- #define I2C_INTENCLR_MONOV (1 << 17)
- #define I2C_INTENCLR_MONIDLE (1 << 19)
- #define I2C_INTENCLR_EVENTTIMEOUT (1 << 24)
- #define I2C_INTENCLR_SCLTIMEOUT (1 << 25)
- #define I2C_TIMEOUT_VAL(n) (((uint32_t) ((n) - 1) & 0xFFF0) | 0x000F)
- #define I2C_INTSTAT_MSTPENDING (1 << 0)
- #define I2C_INTSTAT_MSTRARBLOSS (1 << 4)
- #define I2C_INTSTAT_MSTSTSTPERR (1 << 6)
- #define I2C_INTSTAT_SLVPENDING (1 << 8)
- #define I2C_INTSTAT_SLVNOTSTR (1 << 11)
- #define I2C_INTSTAT_SLVDESEL (1 << 15)
- #define I2C_INTSTAT_MONRDY (1 << 16)
- #define I2C_INTSTAT_MONOV (1 << 17)
- #define I2C_INTSTAT_MONIDLE (1 << 19)
- #define I2C_INTSTAT_EVENTTIMEOUT (1 << 24)
- #define I2C_INTSTAT_SCLTIMEOUT (1 << 25)
- #define I2C_MSTCTL_MSTCONTINUE (1 << 0)
- #define I2C_MSTCTL_MSTSTART (1 << 1)
- #define I2C_MSTCTL_MSTSTOP (1 << 2)
- #define I2C_MSTCTL_MSTDMA (1 << 3)
- #define I2C_MSTTIME_MSTSCLLOW (0x07 << 0)
- #define I2C_MSTTIME_MSTSCLHIGH (0x07 << 4)
- #define I2C_MSTDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_SLVCTL_SLVCONTINUE (1 << 0)
- #define I2C_SLVCTL_SLVNACK (1 << 1)
- #define I2C_SLVCTL_SLVDMA (1 << 3)
- #define I2C_SLVDAT_DATAMASK ((uint32_t) 0x00FF << 0)
- #define I2C_SLVADR_SADISABLE (1 << 0)
- #define I2C_SLVADR_SLVADR (0x7F << 1)
- #define I2C_SLVADR_MASK ((uint32_t) 0x00FF)
- #define I2C_SLVQUAL_QUALMODE0 (1 << 0)
- #define I2C_SLVQUAL_SLVQUAL0 (0x7F << 1)
- #define I2C_MONRXDAT_DATA (0xFF << 0)
- #define I2C_MONRXDAT_MONSTART (1 << 8)
- #define I2C_MONRXDAT_MONRESTART (1 << 9)
- #define I2C_MONRXDAT_MONNACK (1 << 10)

### Functions

- __STATIC_INLINE int Chip_I2C_Init (LPC_I2C_T ∗pI2C)

  *Initialize I2C Interface.*
- __STATIC_INLINE void Chip_I2C_DeInit (LPC_I2C_T ∗pI2C)

  *Shutdown I2C Interface.*
- __STATIC_INLINE void Chip_I2C_SetClockDiv (LPC_I2C_T ∗pI2C, uint32_t clkdiv)

  *Sets I2C Clock Divider registers.*
- static INLINE uint32_t Chip_I2C_GetClockDiv (LPC_I2C_T ∗pI2C)

  *Get I2C Clock Divider registers.*
- static INLINE void Chip_I2C_EnableInt (LPC_I2C_T ∗pI2C, uint32_t intEn)

  *Enable I2C Interrupts.*
- static INLINE void Chip_I2C_DisableInt (LPC_I2C_T ∗pI2C, uint32_t intClr)

  *Disable I2C Interrupts.*
- static INLINE void Chip_I2C_ClearInt (LPC_I2C_T ∗pI2C, uint32_t intClr)

  *Disable I2C Interrupts.*
- static INLINE uint32_t Chip_I2C_GetPendingInt (LPC_I2C_T ∗pI2C)

  *Returns pending I2C Interrupts.*

### Variables

- __IO uint32_t STAT

  *I2C register block structure.*
- __IO uint32_t INTENSET
- __O uint32_t INTENCLR
- __IO uint32_t TIMEOUT
- __IO uint32_t CLKDIV
- __IO uint32_t INTSTAT
- __I uint32_t RESERVED0
- __IO uint32_t MSTCTL
- __IO uint32_t MSTTIME
- __IO uint32_t MSTDAT
- __IO uint32_t RESERVED1 [5]
- __IO uint32_t SLVCTL
- __IO uint32_t SLVDAT
- __IO uint32_t SLVADR [4]
- __IO uint32_t SLVQUAL0
- __IO uint32_t RESERVED2 [9]
- __I uint32_t MONRXDAT
- __IO uint32_t PSELID
- __I uint32_t PID
- LPC_I2C_T

## 8.23 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2cm_5411x.h File Reference

```
#include "chip.h"
```

### Data Structures

- struct I2CM_XFER_T

  *Master transfer data structure definitions.*

## Macros

- #define I2CM_STATUS_OK 0x00
- #define I2CM_STATUS_ERROR 0x01
- #define I2CM_STATUS_NAK_ADR 0x02
- #define I2CM_STATUS_BUS_ERROR 0x03
- #define I2CM_STATUS_NAK_DAT 0x04
- #define I2CM_STATUS_ARBLOST 0x05
- #define I2CM_STATUS_BUSY 0xFF

## Functions

- static INLINE void Chip_I2CM_SetDutyCycle (LPC_I2C_T ∗pI2C, uint16_t sclH, uint16_t sclL)

  *Sets HIGH and LOW duty cycle registers.*
- void Chip_I2CM_SetBusSpeed (LPC_I2C_T ∗pI2C, uint32_t busSpeed)

  *Set up bus speed for LPC_I2C controller.*
- static INLINE void Chip_I2CM_Enable (LPC_I2C_T ∗pI2C)

  *Enable I2C Master interface.*
- static INLINE void Chip_I2CM_Disable (LPC_I2C_T ∗pI2C)

  *Disable I2C Master interface.*
- static INLINE uint32_t Chip_I2CM_GetStatus (LPC_I2C_T ∗pI2C)

  *Get I2C Status.*
- static INLINE void Chip_I2CM_ClearStatus (LPC_I2C_T ∗pI2C, uint32_t clrStatus)

  *Clear I2C status bits (master)*
- static INLINE bool Chip_I2CM_IsMasterPending (LPC_I2C_T ∗pI2C)

  *Check if I2C Master is pending.*
- static INLINE uint32_t Chip_I2CM_GetMasterState (LPC_I2C_T ∗pI2C)

  *Get current state of the I2C Master.*
- static INLINE void Chip_I2CM_SendStart (LPC_I2C_T ∗pI2C)

  *Transmit START or Repeat-START signal on I2C bus.*
- static INLINE void Chip_I2CM_SendStop (LPC_I2C_T ∗pI2C)

  *Transmit STOP signal on I2C bus.*
- static INLINE void Chip_I2CM_MasterContinue (LPC_I2C_T ∗pI2C)

  *Master Continue transfer operation.*
- static INLINE void Chip_I2CM_WriteByte (LPC_I2C_T ∗pI2C, uint8_t data)

  *Transmit a single data byte through the I2C peripheral (master)*
- static INLINE uint8_t Chip_I2CM_ReadByte (LPC_I2C_T ∗pI2C)

  *Read a single byte data from the I2C peripheral (master)*
- uint32_t Chip_I2CM_XferHandler (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transfer state change handler.*
- void Chip_I2CM_Xfer (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transmit and Receive data in master mode.*
- uint32_t Chip_I2CM_XferBlocking (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transmit and Receive data in master mode.*

## 8.24 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2cs_5411x.h File Reference

```
#include "i2c_common_5411x.h"
```

## Data Structures

- struct I2CS_XFER_T

## Typedefs

- typedef void(∗ I2CSlaveXferStart) (uint8_t addr)

    *I2C slave service start callback This callback is called from the I2C slave handler when an I2C slave address is received and needs servicing. It's used to indicate the start of a slave transfer that will happen on the slave bus.*
- typedef uint8_t(∗ I2CSlaveXferSend) (uint8_t ∗data)

    *I2C slave send data callback This callback is called from the I2C slave handler when an I2C slave address needs data to send.*

    *If you want to NAK the master, return I2C_SLVCTL_SLVNACK to the caller. Return I2C_SLVCTL_SLVCONTINUE or 0 to the caller for normal non-DMA data transfer. If you've setup a DMA descriptor for the transfer, return I2C_S←LVCTL_SLVDMA to the caller.*
- typedef uint8_t(∗ I2CSlaveXferRecv) (uint8_t data)

    *I2C slave receive data callback This callback is called from the I2C slave handler when an I2C slave address has receive data.*

    *If you want to NAK the master, return I2C_SLVCTL_SLVNACK to the caller. Return I2C_SLVCTL_SLVCONTINUE or 0 to the caller for normal non-DMA data transfer. If you've setup a DMA descriptor for the transfer, return I2C_S←LVCTL_SLVDMA to the caller.*
- typedef void(∗ I2CSlaveXferDone) (void)

    *I2C slave service done callback This callback is called from the I2C slave handler when an I2C slave transfer is completed. It's used to indicate the end of a slave transfer.*

## Functions

- __STATIC_INLINE void Chip_I2CS_Enable (LPC_I2C_T ∗pI2C)

    *Enable I2C slave interface.*
- __STATIC_INLINE void Chip_I2CS_Disable (LPC_I2C_T ∗pI2C)

    *Disable I2C slave interface.*
- __STATIC_INLINE uint32_t Chip_I2CS_GetStatus (LPC_I2C_T ∗pI2C)

    *Get I2C Status.*
- __STATIC_INLINE void Chip_I2CS_ClearStatus (LPC_I2C_T ∗pI2C, uint32_t clrStatus)

    *Clear I2C status bits (slave)*
- __STATIC_INLINE bool Chip_I2CS_IsSlavePending (LPC_I2C_T ∗pI2C)

    *Check if I2C slave is pending.*
- __STATIC_INLINE bool Chip_I2CS_IsSlaveSelected (LPC_I2C_T ∗pI2C)

    *Check if I2C slave is selected.*
- __STATIC_INLINE bool Chip_I2CS_IsSlaveDeSelected (LPC_I2C_T ∗pI2C)

    *Check if I2C slave is deselected.*
- __STATIC_INLINE uint32_t Chip_I2CS_GetSlaveState (LPC_I2C_T ∗pI2C)

    *Get current state of the I2C slave.*
- __STATIC_INLINE uint32_t Chip_I2CS_GetSlaveMatchIndex (LPC_I2C_T ∗pI2C)

    *Returns the current slave address match index.*
- __STATIC_INLINE void Chip_I2CS_SlaveContinue (LPC_I2C_T ∗pI2C)

    *Slave Continue transfer operation (ACK)*
- __STATIC_INLINE void Chip_I2CS_SlaveNACK (LPC_I2C_T ∗pI2C)

    *Slave NACK operation.*
- __STATIC_INLINE void Chip_I2CS_SlaveEnableDMA (LPC_I2C_T ∗pI2C)

    *Enable slave DMA operation.*

- __STATIC_INLINE void Chip_I2CS_SlaveDisableDMA (LPC_I2C_T ∗pI2C)

    *Disable slave DMA operation.*
- __STATIC_INLINE void Chip_I2CS_WriteByte (LPC_I2C_T ∗pI2C, uint8_t data)

    *Transmit a single data byte through the I2C peripheral (slave)*
- __STATIC_INLINE uint8_t Chip_I2CS_ReadByte (LPC_I2C_T ∗pI2C)

    *Read a single byte data from the I2C peripheral (slave)*
- __STATIC_INLINE void Chip_I2CS_SetSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum, uint8_t slvAddr)

    *Set a I2C slave address for slave operation.*
- __STATIC_INLINE uint8_t Chip_I2CS_GetSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum)

    *Return a I2C programmed slave address.*
- __STATIC_INLINE void Chip_I2CS_EnableSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum)

    *Enable a I2C address.*
- __STATIC_INLINE void Chip_I2CS_DisableSlaveAddr (LPC_I2C_T ∗pI2C, uint8_t slvNum)

    *Disable a I2C address.*
- __STATIC_INLINE void Chip_I2CS_SetSlaveQual0 (LPC_I2C_T ∗pI2C, bool extend, uint8_t slvAddr)

    *Setup slave qialifier address.*
- uint32_t Chip_I2CS_XferHandler (LPC_I2C_T ∗pI2C, const I2CS_XFER_T ∗xfers)

    *Slave transfer state change handler.*

## 8.25 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/i2s_5411x.h File Reference

### Data Structures

- struct LPC_I2S_T

    *I2S register block structure.*
- struct I2S_AUDIO_FORMAT_T
- struct I2S_STATISTICS_T

    *I2S statistics structure.*

### Macros

- #define I2S_CFG1_MAINENABLE (0x01 << 0)

    *I2S CFG1 register bits.*
- #define I2S_CFG1_DATAPAUSE (0x01 << 1)
- #define I2S_CFG1_2NDCOUNT(p) (((p) & 0x03) << 2)
- #define I2S_CFG1_MSTSLVCFG(p) (((p) & 0x03) << 4)
- #define I2S_CFG1_MODE(p) (((p) & 0x03) << 6)
- #define I2S_CFG1_RIGHTLOW (0x01 << 8)
- #define I2S_CFG1_LEFTJUST (0x01 << 9)
- #define I2S_CFG1_ONECHANNEL (0x01 << 10)
- #define I2S_CFG1_PDMDATA (0x01 << 11)
- #define I2S_CFG1_SCK_POL (0x01 << 12)
- #define I2S_CFG1_WS_POL (0x01 << 13)
- #define I2S_CFG1_DATALEN(p) (((p) & 0x1f) << 16)
- #define I2S_CFG2_FRAMELEN(p) (((p) & 0x1ff) << 0)

    *I2S CFG2 register bits.*
- #define I2S_CFG2_POSITION(p) (((p) & 0x1FF) << 16)
- #define I2S_STAT_BUSY (0x01 << 0)

    *I2S status register bits.*
- #define I2S_STAT_SLVFRMERR (0x01 << 1)
- #define I2S_STAT_LR (0x01 << 2)

- #define I2S_STAT_PAUSED (0x01 << 3)
- #define I2S_FIFO_CFG_ENABLETX (0x01 << 0)

  *I2S FIFO configuration bits.*
- #define I2S_FIFO_CFG_ENABLERX (0x01 << 1)
- #define I2S_FIFO_CFG_TXI2SE0 (0x01 << 2)
- #define I2S_FIFO_CFG_PACK48 (0x01 << 3)
- #define I2S_FIFO_CFG_SIZE(p) (((p) & 0x03) << 4)
- #define I2S_FIFO_CFG_DMATX (0x01 << 12)
- #define I2S_FIFO_CFG_DMARX (0x01 << 13)
- #define I2S_FIFO_CFG_WAKETX (0x01 << 14)
- #define I2S_FIFO_CFG_WAKERX (0x01 << 15)
- #define I2S_FIFO_CFG_EMPTYTX (0x01 << 16)
- #define I2S_FIFO_CFG_EMPTYRX (0x01 << 17)
- #define I2S_FIFO_CFG_POPDBG (0x01 << 18)
- #define I2S_FIFO_STAT_TXERR (0x01 << 0)

  *I2S FIFO status bits.*
- #define I2S_FIFO_STAT_RXERR (0x01 << 1)
- #define I2S_FIFO_STAT_PERINT (0x01 << 3)
- #define I2S_FIFO_STAT_TXEMPTY (0x01 << 4)
- #define I2S_FIFO_STAT_TXNOTFULL (0x01 << 5)
- #define I2S_FIFO_STAT_RXNOTEMPTY (0x01 << 6)
- #define I2S_FIFO_STAT_RXFULL (0x01 << 7)
- #define I2S_FIFO_STAT_TXLVL(p) (((p) & 0x0f) << 8)
- #define I2S_FIFO_STAT_RXLVL(p) (((p) & 0x0f) << 16)
- #define I2S_FIFO_TRIG_TXLVLENA (0x01 << 0)

  *I2S FIFO trigger settings bits.*
- #define I2S_FIFO_TRIG_RXLVLENA (0x01 << 1)
- #define I2S_FIFO_TRIG_TXLVL(p) (((p) & 0x0f) << 8)
- #define I2S_FIFO_TRIG_RXLVL(p) (((p) & 0x0f) << 16)
- #define I2S_FIFO_INT_BITMASK (0x001F) /∗∗ FIFO interrupt Bit mask ∗/

  *I2S FIFO interrupt enable. Used for interrupt status too.*
- #define I2S_FIFO_INT_TXERR (0x01 << 0)
- #define I2S_FIFO_INT_RXERR (0x01 << 1)
- #define I2S_FIFO_INT_TXLVL (0x01 << 2)
- #define I2S_FIFO_INT_RXLVL (0x01 << 3)
- #define I2S_FIFO_INT_PERINT (0x01 << 4)
- #define LPC_I2S6_BASE __APPEND3(LPC_FLEXCOMM,I2S6_FLEXCOMM,_BASE)
- #define LPC_I2S6 ((LPC_I2S_T ∗) LPC_I2S6_BASE)
- #define I2S6_IRQHandler __APPEND3(FLEXCOMM,I2S6_FLEXCOMM,_IRQHandler)
- #define I2S6_IRQn __APPEND3(FLEXCOMM,I2S6_FLEXCOMM,_IRQn)
- #define DMAREQ_I2S6_RX __APPEND3(DMAREQ_FLEXCOMM,I2S6_FLEXCOMM,_RX)
- #define DMAREQ_I2S6_TX __APPEND3(DMAREQ_FLEXCOMM,I2S6_FLEXCOMM,_TX)
- #define LPC_I2S7_BASE __APPEND3(LPC_FLEXCOMM,I2S7_FLEXCOMM,_BASE)
- #define LPC_I2S7 ((LPC_I2S_T ∗) LPC_I2S7_BASE)
- #define I2S7_IRQHandler __APPEND3(FLEXCOMM,I2S7_FLEXCOMM,_IRQHandler)
- #define I2S7_IRQn __APPEND3(FLEXCOMM,I2S7_FLEXCOMM,_IRQn)
- #define DMAREQ_I2S7_RX __APPEND3(DMAREQ_FLEXCOMM,I2S7_FLEXCOMM,_RX)
- #define DMAREQ_I2S7_TX __APPEND3(DMAREQ_FLEXCOMM,I2S7_FLEXCOMM,_TX)

## Enumerations

- enum I2S_FIFO_CMD_T {
I2S_FIFO_ENABLE, I2S_FIFO_DISABLE, I2S_FIFO_DMA_ENABLE, I2S_FIFO_DMA_DISABLE,
I2S_FIFO_CLEAR, I2S_FIFO_TXZ_ENABLE, I2S_FIFO_TXZ_DISABLE }
- enum I2S_DIR_T { I2S_TX, I2S_RX }
- enum I2S_MSTSLVCFG_T { NORMAL_SLAVE, WS_SYNC_MASTER, EXT_SCLCK_MASTER, NORMA↩
L_MASTER }
- enum I2S_MODE_T { I2S_CLASSIC, DSP_WS_50, DSP_WS_SHORT, DSP_WS_LONG }

## Functions

- int Chip_I2S_Init (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt)

  *Initialize I2S driver.*
- __STATIC_INLINE int Chip_I2S_TX_Init (LPC_I2S_T ∗pI2S)
- __STATIC_INLINE int Chip_I2S_RX_Init (LPC_I2S_T ∗pI2S)
- __STATIC_INLINE void Chip_I2S_DeInit (LPC_I2S_T ∗pI2S)

  *Shutdown I2S driver.*
- Status Chip_I2S_Config (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt)

  *Configure I2S Port.*
- __STATIC_INLINE uint32_t Chip_I2S_GetStatus (LPC_I2S_T ∗pI2S)

  *Get the I2S status register.*
- __STATIC_INLINE void Chip_I2S_ClearStatus (LPC_I2S_T ∗pI2S, uint32_t stsMask)

  *Clear the I2S status register.*
- __STATIC_INLINE void Chip_I2S_Send (LPC_I2S_T ∗pI2S, uint32_t data)

  *Send a 32-bit data to TXFIFO for transmition.*
- __STATIC_INLINE uint32_t Chip_I2S_Receive (LPC_I2S_T ∗pI2S)

  *Get received data from RXFIFO.*
- __STATIC_INLINE void Chip_I2S_Start (LPC_I2S_T ∗pI2S)

  *Start I2S port.*
- __STATIC_INLINE void Chip_I2S_Stop (LPC_I2S_T ∗pI2S)

  *Stop I2S asynchronously.*
- __STATIC_INLINE void Chip_I2S_Pause (LPC_I2S_T ∗pI2S)

  *Pause the I2S port.*
- __STATIC_INLINE void Chip_I2S_Play (LPC_I2S_T ∗pI2S)

  *Play (un-Pause) the I2S port.*
- void Chip_I2S_FIFO_Config (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt)

  *Configure I2S FIFO.*
- void Chip_I2S_FIFO_Control (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt, I2S_FIFO_CMD_T cmd)

  *Execute I2S FIFO control commands.*
- __STATIC_INLINE uint32_t Chip_I2S_GetFIFOStatus (LPC_I2S_T ∗pI2S)

  *Get FIFO status.*
- __STATIC_INLINE void Chip_I2S_ClrFIFOStatus (LPC_I2S_T ∗pI2S, uint32_t mask)

  *Clear FIFO status.*
- __STATIC_INLINE void Chip_I2S_SetFIFOTrigLevel (LPC_I2S_T ∗pI2S, uint8_t tx_lvl, uint8_t rx_lvl)

  *Setup I2S FIFO trigger-level.*
- __STATIC_INLINE uint32_t Chip_I2S_GetFIFOTrigLevel (LPC_I2S_T ∗pI2S)

  *Get I2S FIFO trigger-level.*
- __STATIC_INLINE uint8_t Chip_I2S_GetFIFOTxLevel (LPC_I2S_T ∗pI2S)

  *Get the current level of the Transmit FIFO.*
- __STATIC_INLINE uint8_t Chip_I2S_GetFIFORxLevel (LPC_I2S_T ∗pI2S)

*Get the current level of the Receive FIFO.*

- __STATIC_INLINE void Chip_I2S_FIFO_SetInterrupt (LPC_I2S_T ∗pI2S, uint32_t int_val)

  *Set I2S FIFO interrupts.*
- __STATIC_INLINE void Chip_I2S_FIFO_ClrInterrupt (LPC_I2S_T ∗pI2S, uint32_t int_val)

  *Clear I2S FIFO interrupts.*
- __STATIC_INLINE uint32_t Chip_I2S_FIFO_GetPendingInts (LPC_I2S_T ∗pI2S)

  *Get I2S FIFO interrupts status.*
- __STATIC_INLINE void Chip_I2S_FIFO_ClearStatus (LPC_I2S_T ∗pI2S, uint32_t mask)

  *Clear the FIFO status register.*
- void Chip_I2S_ErrorHandler (LPC_I2S_T ∗pI2S, I2S_STATISTICS_T ∗stat)

  *I2S error handler.*

## 8.25.1 Macro Definition Documentation

### 8.25.1.1 #define DMAREQ_I2S6_RX __APPEND3(DMAREQ_FLEXCOMM,I2S6_FLEXCOMM,_RX)

Definition at line 506 of file i2s_5411x.h.

### 8.25.1.2 #define DMAREQ_I2S6_TX __APPEND3(DMAREQ_FLEXCOMM,I2S6_FLEXCOMM,_TX)

Definition at line 507 of file i2s_5411x.h.

### 8.25.1.3 #define DMAREQ_I2S7_RX __APPEND3(DMAREQ_FLEXCOMM,I2S7_FLEXCOMM,_RX)

Definition at line 514 of file i2s_5411x.h.

### 8.25.1.4 #define DMAREQ_I2S7_TX __APPEND3(DMAREQ_FLEXCOMM,I2S7_FLEXCOMM,_TX)

Definition at line 515 of file i2s_5411x.h.

### 8.25.1.5 #define I2S6_IRQHandler __APPEND3(FLEXCOMM,I2S6_FLEXCOMM,_IRQHandler)

Definition at line 504 of file i2s_5411x.h.

### 8.25.1.6 #define I2S6_IRQn __APPEND3(FLEXCOMM,I2S6_FLEXCOMM,_IRQn)

Definition at line 505 of file i2s_5411x.h.

### 8.25.1.7 #define I2S7_IRQHandler __APPEND3(FLEXCOMM,I2S7_FLEXCOMM,_IRQHandler)

Definition at line 512 of file i2s_5411x.h.

### 8.25.1.8 #define I2S7_IRQn __APPEND3(FLEXCOMM,I2S7_FLEXCOMM,_IRQn)

Definition at line 513 of file i2s_5411x.h.

### 8.25.1.9 #define I2S_CFG1_2NDCOUNT( *p* ) (((p) & 0x03) << 2)

additional I2S channel pair ct (read only)

Definition at line 80 of file i2s_5411x.h.

#### 8.25.1.10 #define I2S_CFG1_DATALEN( *p* ) (((p) & 0x1f) << 16)

Data Length (minus 1 encoded)

Definition at line 89 of file i2s_5411x.h.

#### 8.25.1.11 #define I2S_CFG1_DATAPAUSE (0x01 << 1)

Data flow Pause

Definition at line 79 of file i2s_5411x.h.

#### 8.25.1.12 #define I2S_CFG1_LEFTJUST (0x01 << 9)

Left Justify data

Definition at line 84 of file i2s_5411x.h.

#### 8.25.1.13 #define I2S_CFG1_MAINENABLE (0x01 << 0)

I2S CFG1 register bits.

I2S Main enable

Definition at line 78 of file i2s_5411x.h.

#### 8.25.1.14 #define I2S_CFG1_MODE( *p* ) (((p) & 0x03) << 6)

Selects basic I2S operating mode

Definition at line 82 of file i2s_5411x.h.

#### 8.25.1.15 #define I2S_CFG1_MSTSLVCFG( *p* ) (((p) & 0x03) << 4)

Master / slave configuration selection

Definition at line 81 of file i2s_5411x.h.

#### 8.25.1.16 #define I2S_CFG1_ONECHANNEL (0x01 << 10)

Single channel mode

Definition at line 85 of file i2s_5411x.h.

#### 8.25.1.17 #define I2S_CFG1_PDMDATA (0x01 << 11)

PDM Data selection

Definition at line 86 of file i2s_5411x.h.

#### 8.25.1.18 #define I2S_CFG1_RIGHTLOW (0x01 << 8)

Right channel data is in the Low portion of FIFO data

Definition at line 83 of file i2s_5411x.h.

#### 8.25.1.19 #define I2S_CFG1_SCK_POL (0x01 << 12)

SCK polarity

Definition at line 87 of file i2s_5411x.h.

#### 8.25.1.20 #define I2S_CFG1_WS_POL (0x01 << 13)

WS polarity

Definition at line 88 of file i2s_5411x.h.

#### 8.25.1.21 #define I2S_CFG2_FRAMELEN( *p* ) (((p) & 0x1ff) << 0)

I2S CFG2 register bits.

I2S Main enable

Definition at line 94 of file i2s_5411x.h.

#### 8.25.1.22 #define I2S_CFG2_POSITION( *p* ) (((p) & 0x1FF) << 16)

Data flow Pause

Definition at line 95 of file i2s_5411x.h.

#### 8.25.1.23 #define I2S_FIFO_CFG_DMARX (0x01 << 13)

DMA configuration for receive

Definition at line 114 of file i2s_5411x.h.

#### 8.25.1.24 #define I2S_FIFO_CFG_DMATX (0x01 << 12)

DMA configuration for transmit

Definition at line 113 of file i2s_5411x.h.

#### 8.25.1.25 #define I2S_FIFO_CFG_EMPTYRX (0x01 << 17)

Empty command for the receive FIFO

Definition at line 118 of file i2s_5411x.h.

#### 8.25.1.26 #define I2S_FIFO_CFG_EMPTYTX (0x01 << 16)

Empty command for the transmit FIFO

Definition at line 117 of file i2s_5411x.h.

#### 8.25.1.27 #define I2S_FIFO_CFG_ENABLERX (0x01 << 1)

Enable the receive FIFO

Definition at line 109 of file i2s_5411x.h.

#### 8.25.1.28 #define I2S_FIFO_CFG_ENABLETX (0x01 $<<$ 0)

I2S FIFO configuration bits.

Enable the transmit FIFO

Definition at line 108 of file i2s_5411x.h.

#### 8.25.1.29 #define I2S_FIFO_CFG_PACK48 (0x01 $<<$ 3)

Packing format for 48-bit data

Definition at line 111 of file i2s_5411x.h.

#### 8.25.1.30 #define I2S_FIFO_CFG_POPDBG (0x01 $<<$ 18)

Pop FIFO for debug reads

Definition at line 119 of file i2s_5411x.h.

#### 8.25.1.31 #define I2S_FIFO_CFG_SIZE( *p* ) (((p) & 0x03) $<<$ 4)

FIFO size (READ ONLY)

Definition at line 112 of file i2s_5411x.h.

#### 8.25.1.32 #define I2S_FIFO_CFG_TXI2SE0 (0x01 $<<$ 2)

Transmit I2S empty 0

Definition at line 110 of file i2s_5411x.h.

#### 8.25.1.33 #define I2S_FIFO_CFG_WAKERX (0x01 $<<$ 15)

Wake-up for receive FIFO level

Definition at line 116 of file i2s_5411x.h.

#### 8.25.1.34 #define I2S_FIFO_CFG_WAKETX (0x01 $<<$ 14)

Wake-up for transmit FIFO level

Definition at line 115 of file i2s_5411x.h.

#### 8.25.1.35 #define I2S_FIFO_INT_BITMASK (0x001F) /$**$ FIFO interrupt Bit mask $*$/

I2S FIFO interrupt enable. Used for interrupt status too.

Definition at line 146 of file i2s_5411x.h.

#### 8.25.1.36 #define I2S_FIFO_INT_PERINT (0x01 $<<$ 4)

I2S peripheral interrupt [BIT-4 of FIFOINTSTAT register]

Definition at line 151 of file i2s_5411x.h.

#### 8.25.1.37 #define I2S_FIFO_INT_RXERR (0x01 $<<$ 1)

Interrupt on RX error

Definition at line 148 of file i2s_5411x.h.

#### 8.25.1.38 #define I2S_FIFO_INT_RXLVL (0x01 $<<$ 3)

Interrupt on RX level

Definition at line 150 of file i2s_5411x.h.

#### 8.25.1.39 #define I2S_FIFO_INT_TXERR (0x01 $<<$ 0)

Interrupt on TX error

Definition at line 147 of file i2s_5411x.h.

#### 8.25.1.40 #define I2S_FIFO_INT_TXLVL (0x01 $<<$ 2)

Interrupt on TX level

Definition at line 149 of file i2s_5411x.h.

#### 8.25.1.41 #define I2S_FIFO_STAT_PERINT (0x01 $<<$ 3)

Peripheral interrupt

Definition at line 126 of file i2s_5411x.h.

#### 8.25.1.42 #define I2S_FIFO_STAT_RXERR (0x01 $<<$ 1)

RX FIFO error

Definition at line 125 of file i2s_5411x.h.

#### 8.25.1.43 #define I2S_FIFO_STAT_RXFULL (0x01 $<<$ 7)

Receive FIFO full

Definition at line 130 of file i2s_5411x.h.

#### 8.25.1.44 #define I2S_FIFO_STAT_RXLVL( *p* ) (((p) & 0x0f) $<<$ 16)

Receive FIFO current level

Definition at line 132 of file i2s_5411x.h.

#### 8.25.1.45 #define I2S_FIFO_STAT_RXNOTEMPTY (0x01 $<<$ 6)

Receive FIFO not empty

Definition at line 129 of file i2s_5411x.h.

#### 8.25.1.46 #define I2S_FIFO_STAT_TXEMPTY (0x01 << 4)

Transmit FIFO empty

Definition at line 127 of file i2s_5411x.h.

#### 8.25.1.47 #define I2S_FIFO_STAT_TXERR (0x01 << 0)

I2S FIFO status bits.

TX FIFO error

Definition at line 124 of file i2s_5411x.h.

#### 8.25.1.48 #define I2S_FIFO_STAT_TXLVL( *p* ) (((p) & 0x0f) << 8)

Transmit FIFO current level

Definition at line 131 of file i2s_5411x.h.

#### 8.25.1.49 #define I2S_FIFO_STAT_TXNOTFULL (0x01 << 5)

Transmit FIFO not full

Definition at line 128 of file i2s_5411x.h.

#### 8.25.1.50 #define I2S_FIFO_TRIG_RXLVL( *p* ) (((p) & 0x0f) << 16)

Receive FIFO level trigger point

Definition at line 140 of file i2s_5411x.h.

#### 8.25.1.51 #define I2S_FIFO_TRIG_RXLVLENA (0x01 << 1)

Receive FIFO level trigger enable

Definition at line 138 of file i2s_5411x.h.

#### 8.25.1.52 #define I2S_FIFO_TRIG_TXLVL( *p* ) (((p) & 0x0f) << 8)

Transmit FIFO level trigger point

Definition at line 139 of file i2s_5411x.h.

#### 8.25.1.53 #define I2S_FIFO_TRIG_TXLVLENA (0x01 << 0)

I2S FIFO trigger settings bits.

Transmit FIFO level trigger enable

Definition at line 137 of file i2s_5411x.h.

#### 8.25.1.54 #define I2S_STAT_BUSY (0x01 << 0)

I2S status register bits.

Busy status for the primary channel pair

Definition at line 100 of file i2s_5411x.h.

#### 8.25.1.55 #define I2S_STAT_LR (0x01 << 2)

Left/Right indication

Definition at line 102 of file i2s_5411x.h.

#### 8.25.1.56 #define I2S_STAT_PAUSED (0x01 << 3)

Data Paused status flag

Definition at line 103 of file i2s_5411x.h.

#### 8.25.1.57 #define I2S_STAT_SLVFRMERR (0x01 << 1)

Slave Frame Error flag

Definition at line 101 of file i2s_5411x.h.

#### 8.25.1.58 #define LPC_I2S6 ((LPC_I2S_T *) LPC_I2S6_BASE)

Definition at line 503 of file i2s_5411x.h.

#### 8.25.1.59 #define LPC_I2S6_BASE __APPEND3(LPC_FLEXCOMM,I2S6_FLEXCOMM,_BASE)

Definition at line 502 of file i2s_5411x.h.

#### 8.25.1.60 #define LPC_I2S7 ((LPC_I2S_T *) LPC_I2S7_BASE)

Definition at line 511 of file i2s_5411x.h.

#### 8.25.1.61 #define LPC_I2S7_BASE __APPEND3(LPC_FLEXCOMM,I2S7_FLEXCOMM,_BASE)

Definition at line 510 of file i2s_5411x.h.

### 8.25.2 Enumeration Type Documentation

#### 8.25.2.1 enum I2S_DIR_T

Enumerator

***I2S_TX***

***I2S_RX***

Definition at line 163 of file i2s_5411x.h.

#### 8.25.2.2 enum I2S_FIFO_CMD_T

Enumerator

***I2S_FIFO_ENABLE***

***I2S_FIFO_DISABLE***

***I2S_FIFO_DMA_ENABLE***

***I2S_FIFO_DMA_DISABLE***

***I2S_FIFO_CLEAR***

***I2S_FIFO_TXZ_ENABLE***

***I2S_FIFO_TXZ_DISABLE***

Definition at line 153 of file i2s_5411x.h.

#### 8.25.2.3 enum I2S_MODE_T

**Enumerator**

***I2S_CLASSIC***

***DSP_WS_50***

***DSP_WS_SHORT***

***DSP_WS_LONG***

Definition at line 175 of file i2s_5411x.h.

#### 8.25.2.4 enum I2S_MSTSLVCFG_T

**Enumerator**

***NORMAL_SLAVE***

***WS_SYNC_MASTER***

***EXT_SCLCK_MASTER***

***NORMAL_MASTER***

Definition at line 168 of file i2s_5411x.h.

### 8.25.3 Function Documentation

#### 8.25.3.1 __STATIC_INLINE void Chip_I2S_ClearStatus ( LPC_I2S_T ∗ *pI2S,* uint32_t *stsMask* )

Clear the I2S status register.

**Parameters**

| | |
|---|---|
| *pI2S* | : Pointer to selected I2Sx peripheral |
| *stsMask* | : OR'ed statuses to disable |

**Returns**

Nothing

**Note**

Multiple interrupts may be pending. Mask the return value with one or more I2S_INTEN_∗ definitions to determine pending interrupts.

Definition at line 272 of file i2s_5411x.h.

#### 8.25.3.2 __STATIC_INLINE void Chip_I2S_ClrFIFOStatus ( LPC_I2S_T ∗ *pI2S,* uint32_t *mask* )

Clear FIFO status.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |
| *mask* | : Mask of the status bits that needs to be cleared |

**Returns**

mothing

Definition at line 388 of file i2s_5411x.h.

#### 8.25.3.3 Status Chip_I2S_Config ( LPC_I2S_T ∗ *pI2S,* I2S_AUDIO_FORMAT_T ∗ *fmt* )

Configure I2S Port.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *fmt* | : Audio Format |

**Returns**

SUCCESS or ERROR

Definition at line 67 of file i2s_5411x.c.

#### 8.25.3.4 __STATIC_INLINE void Chip_I2S_DeInit ( LPC_I2S_T ∗ *pI2S* )

Shutdown I2S driver.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Nothing

**Note**

Reset all relative registers (DMA, transmit/receive control, interrupt) to default value

Definition at line 237 of file i2s_5411x.h.

#### 8.25.3.5 void Chip_I2S_ErrorHandler ( LPC_I2S_T ∗ *pI2S,* I2S_STATISTICS_T ∗ *stat* )

I2S error handler.

**Parameters**

| | |
|---|---|
| *I2S* | : The base of the I2S peripheral on the chip |
| *stat* | : Statistics structure |

**Returns**

Nothing

Definition at line 210 of file i2s_5411x.c.

#### 8.25.3.6 __STATIC_INLINE void Chip_I2S_FIFO_ClearStatus ( LPC_I2S_T ∗ *pI2S,* uint32_t *mask* )

Clear the FIFO status register.

**Parameters**

| | |
|---|---|
| *I2S* | : The base of the I2S peripheral on the chip |
| *mask* | : Mask of the status bits that needs to be cleared |

**Returns**

Nothing

Definition at line 487 of file i2s_5411x.h.

#### 8.25.3.7 __STATIC_INLINE void Chip_I2S_FIFO_ClrInterrupt ( LPC_I2S_T ∗ *pI2S*, uint32_t *int_val* )

Clear I2S FIFO interrupts.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *int_val* | : Interrupts to clear |

**Returns**

Nothing

Definition at line 466 of file i2s_5411x.h.

#### 8.25.3.8 void Chip_I2S_FIFO_Config ( LPC_I2S_T ∗ *pI2S*, I2S_AUDIO_FORMAT_T ∗ *fmt* )

Configure I2S FIFO.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *fmpt* | : Audio format information |

**Returns**

Nothing

Definition at line 120 of file i2s_5411x.c.

#### 8.25.3.9 void Chip_I2S_FIFO_Control ( LPC_I2S_T ∗ *pI2S*, I2S_AUDIO_FORMAT_T ∗ *fmt*, I2S_FIFO_CMD_T *cmd* )

Execute I2S FIFO control commands.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *fmt* | : Audio format information |
| *cmd* | : FIFO command |

**Returns**

Nothing

Definition at line 191 of file i2s_5411x.c.

#### 8.25.3.10 __STATIC_INLINE uint32_t Chip_I2S_FIFO_GetPendingInts ( LPC_I2S_T ∗ *pI2S* )

Get I2S FIFO interrupts status.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |

**Returns**

interrupt status

Definition at line 476 of file i2s_5411x.h.

#### 8.25.3.11 __STATIC_INLINE void Chip_I2S_FIFO_SetInterrupt ( LPC_I2S_T ∗ *pI2S,* uint32_t *int_val* )

Set I2S FIFO interrupts.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *int_val* | : Interrupts to set |

**Returns**

Nothing

Definition at line 455 of file i2s_5411x.h.

#### 8.25.3.12 __STATIC_INLINE uint8_t Chip_I2S_GetFIFORxLevel ( LPC_I2S_T ∗ *pI2S* )

Get the current level of the Receive FIFO.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Current level of the Receive FIFO

Definition at line 441 of file i2s_5411x.h.

#### 8.25.3.13 __STATIC_INLINE uint32_t Chip_I2S_GetFIFOStatus ( LPC_I2S_T ∗ *pI2S* )

Get FIFO status.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Current FIFO status

Definition at line 377 of file i2s_5411x.h.

#### 8.25.3.14 __STATIC_INLINE uint32_t Chip_I2S_GetFIFOTrigLevel ( LPC_I2S_T ∗ *pI2S* )

Get I2S FIFO trigger-level.

**Parameters**

| | |
|---|---|
| *pI2S* | : Base of on-chip I2S peripheral |

**Returns**

Returns the complete raw trigger register.

Definition at line 418 of file i2s_5411x.h.

#### 8.25.3.15 __STATIC_INLINE uint8_t Chip_I2S_GetFIFOTxLevel ( LPC_I2S_T ∗ *pI2S* )

Get the current level of the Transmit FIFO.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Current level of the Transmit FIFO

Definition at line 431 of file i2s_5411x.h.

#### 8.25.3.16 __STATIC_INLINE uint32_t Chip_I2S_GetStatus ( LPC_I2S_T ∗ *pI2S* )

Get the I2S status register.

**Parameters**

| | |
|---|---|
| *pI2S* | : Pointer to selected I2Sx peripheral |

**Returns**

I2S status register

**Note**

Multiple statuses may be pending. Mask the return value with one or more I2S_STAT_∗ definitions to determine statuses.

Definition at line 258 of file i2s_5411x.h.

#### 8.25.3.17 int Chip_I2S_Init ( LPC_I2S_T ∗ *pI2S,* I2S_AUDIO_FORMAT_T ∗ *fmt* )

Initialize I2S driver.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

0 - on success; ERR_FLEXCOMM_FUNCNOTSUPPORTED or ERR_FLEXCOMM_NOTFREE on failure

Definition at line 49 of file i2s_5411x.c.

#### 8.25.3.18 __STATIC_INLINE void Chip_I2S_Pause ( LPC_I2S_T ∗ *pI2S* )

Pause the I2S port.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Nothing

Definition at line 326 of file i2s_5411x.h.

#### 8.25.3.19 __STATIC_INLINE void Chip_I2S_Play ( LPC_I2S_T ∗ *pI2S* )

Play (un-Pause) the I2S port.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Nothing

Definition at line 336 of file i2s_5411x.h.

#### 8.25.3.20 __STATIC_INLINE uint32_t Chip_I2S_Receive ( LPC_I2S_T ∗ *pI2S* )

Get received data from RXFIFO.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Data received in RXFIFO

**Note**

The function reads from RXFIFO without checking any condition.

Definition at line 295 of file i2s_5411x.h.

#### 8.25.3.21 __STATIC_INLINE int Chip_I2S_RX_Init ( LPC_I2S_T ∗ *pI2S* )

Definition at line 226 of file i2s_5411x.h.

#### 8.25.3.22 __STATIC_INLINE void Chip_I2S_Send ( LPC_I2S_T ∗ *pI2S,* uint32_t *data* )

Send a 32-bit data to TXFIFO for transmition.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

| | |
|---|---|
| *data* | : Data to be transmited |

**Returns**

Nothing

**Note**

The function writes to TXFIFO without checking any condition.

Definition at line 284 of file i2s_5411x.h.

#### 8.25.3.23 __STATIC_INLINE void Chip_I2S_SetFIFOTrigLevel ( LPC_I2S_T ∗ *pI2S*, uint8_t *tx_lvl*, uint8_t *rx_lvl* )

Setup I2S FIFO trigger-level.

**Parameters**

| | |
|---|---|
| *pI2S* | : Base on-chip I2S peripheral address |
| *tx_lvl* | : TX Trigger level [Valid values 0 to 7] |
| *rx_lvl* | : RX Trigger level [Valid values 0 to 7] |

**Returns**

Nothing

**Note**

When *tx_lvl* = 0; trigger will happen when TX FIFO is empty if *tx_lvl* = 7; trigger will happen when TX FIFO has at least one free space
When *rx_lvl* = 0; trigger will happen when RX FIFO has at least one data in it, if *rx_lvl* = 7; trigger will happen when RX FIFO is full and cannot receive anymore data.

Definition at line 408 of file i2s_5411x.h.

#### 8.25.3.24 __STATIC_INLINE void Chip_I2S_Start ( LPC_I2S_T ∗ *pI2S* )

Start I2S port.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Nothing

Definition at line 305 of file i2s_5411x.h.

#### 8.25.3.25 __STATIC_INLINE void Chip_I2S_Stop ( LPC_I2S_T ∗ *pI2S* )

Stop I2S asynchronously.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

Nothing

**Note**

Pause, resets the transmit channel and FIFO asynchronously

Definition at line 316 of file i2s_5411x.h.

#### 8.25.3.26 __STATIC_INLINE int Chip_I2S_TX_Init ( LPC_I2S_T ∗ *pI2S* )

Definition at line 222 of file i2s_5411x.h.

## 8.26 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/iap.h File Reference

### Macros

- #define IAP_PREWRRITE_CMD 50
- #define IAP_WRISECTOR_CMD 51
- #define IAP_ERSSECTOR_CMD 52
- #define IAP_BLANK_CHECK_SECTOR_CMD 53
- #define IAP_REPID_CMD 54
- #define IAP_READ_BOOT_CODE_CMD 55
- #define IAP_COMPARE_CMD 56
- #define IAP_REINVOKE_ISP_CMD 57
- #define IAP_READ_UID_CMD 58
- #define IAP_ERASE_PAGE_CMD 59
- #define IAP_EEPROM_WRITE 61
- #define IAP_EEPROM_READ 62
- #define IAP_CMD_SUCCESS 0
- #define IAP_INVALID_COMMAND 1
- #define IAP_SRC_ADDR_ERROR 2
- #define IAP_DST_ADDR_ERROR 3
- #define IAP_SRC_ADDR_NOT_MAPPED 4
- #define IAP_DST_ADDR_NOT_MAPPED 5
- #define IAP_COUNT_ERROR 6
- #define IAP_INVALID_SECTOR 7
- #define IAP_SECTOR_NOT_BLANK 8
- #define IAP_SECTOR_NOT_PREPARED 9
- #define IAP_COMPARE_ERROR 10
- #define IAP_BUSY 11
- #define IAP_PARAM_ERROR 12
- #define IAP_ADDR_ERROR 13
- #define IAP_ADDR_NOT_MAPPED 14
- #define IAP_CMD_LOCKED 15
- #define IAP_INVALID_CODE 16
- #define IAP_INVALID_BAUD_RATE 17
- #define IAP_INVALID_STOP_BIT 18
- #define IAP_CRP_ENABLED 19

### Typedefs

- typedef void(∗ IAP_ENTRY_T) (unsigned int[5], unsigned int[4])

### Functions

- uint8_t Chip_IAP_PreSectorForReadWrite (uint32_t strSector, uint32_t endSector)

  *Prepare sector for write operation.*
- uint8_t Chip_IAP_CopyRamToFlash (uint32_t dstAdd, uint32_t ∗srcAdd, uint32_t byteswrt)

  *Copy RAM to flash.*
- uint8_t Chip_IAP_EraseSector (uint32_t strSector, uint32_t endSector)

  *Erase sector.*
- uint8_t Chip_IAP_BlankCheckSector (uint32_t strSector, uint32_t endSector)

  *Blank check a sector or multiples sector of on-chip flash memory.*
- uint32_t Chip_IAP_ReadPID (void)

  *Read part identification number.*
- uint8_t Chip_IAP_ReadBootCode (void)

  *Read boot code version number.*
- uint8_t Chip_IAP_Compare (uint32_t dstAdd, uint32_t srcAdd, uint32_t bytescmp)

  *Compare the memory contents at two locations.*
- uint8_t Chip_IAP_ReinvokeISP (void)

  *IAP reinvoke ISP to invoke the bootloader in ISP mode.*
- uint32_t Chip_IAP_ReadUID (void)

  *Read the unique ID.*
- uint8_t Chip_IAP_ErasePage (uint32_t strPage, uint32_t endPage)

  *Erase a page or multiple papers of on-chip flash memory.*

## 8.27 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/inmux_5411x.h File Reference

### Data Structures

- struct LPC_INMUX_T

  *LPC5411X Input Mux Register Block Structure.*

### Enumerations

- enum DMA_TRIGSRC_T {
  DMATRIG_ADC0_SEQA_IRQ = 0, DMATRIG_ADC0_SEQB_IRQ, DMATRIG_SCT0_DMA0, DMATRIG_SCT0_DMA1,
  DMATRIG_TIMER0_MATCH0, DMATRIG_TIMER0_MATCH1, DMATRIG_TIMER1_MATCH0, DMATRIG_TIMER2_MATCH0,
  DMATRIG_TIMER2_MATCH1, DMATRIG_TIMER3_MATCH0, DMATRIG_TIMER4_MATCH0, DMATRIG_TIMER4_MATCH1,
  DMATRIG_PININT0, DMATRIG_PININT1, DMATRIG_PININT2, DMATRIG_PININT3,
  DMATRIG_OUTMUX0, DMATRIG_OUTMUX1, DMATRIG_OUTMUX2, DMATRIG_OUTMUX3 }
- enum FREQMSR_SRC_T {
  FREQMSR_CLKIN = 0, FREQMSR_FRO12MHZ, FREQMSR_WDOSC, FREQMSR_32KHZOSC,
  FREQ_MEAS_MAIN_CLK, FREQMSR_PIO0_4, FREQMSR_PIO0_20, FREQMSR_PIO0_24,
  FREQMSR_PIO1_4 }

### Functions

- __STATIC_INLINE void Chip_INMUX_PinIntSel (uint8_t pintSel, uint8_t portNum, uint8_t pinNum)

  *GPIO Pin Interrupt Pin Select (sets PINTSEL register)*
- __STATIC_INLINE void Chip_INMUX_SetDMATrigger (uint8_t ch, DMA_TRIGSRC_T trig)

  *Select a trigger source for a DMA channel.*
- __STATIC_INLINE void Chip_INMUX_SetDMAOutMux (uint8_t index, uint8_t dmaCh)

  *Selects a DMA trigger source for the DMATRIG_OUTMUXn IDs.*
- __STATIC_INLINE void Chip_INMUX_SetFreqMeasRefClock (FREQMSR_SRC_T ref)

  *Selects a reference clock used with the frequency measure function.*
- __STATIC_INLINE void Chip_INMUX_SetFreqMeasTargClock (FREQMSR_SRC_T targ)

  *Selects a target clock used with the frequency measure function.*

## 8.28 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/iocon_5411x.h File Reference

### Data Structures

- struct LPC_IOCON_T

  *LPC5411X IO Configuration Unit register block structure.*
- struct PINMUX_GRP_T

  *Array of IOCON pin definitions passed to Chip_IOCON_SetPinMuxing() must be in this format.*

### Macros

- #define IOCON_FUNC0 0x0
- #define IOCON_FUNC1 0x1
- #define IOCON_FUNC2 0x2
- #define IOCON_FUNC3 0x3
- #define IOCON_FUNC4 0x4
- #define IOCON_FUNC5 0x5
- #define IOCON_FUNC6 0x6
- #define IOCON_FUNC7 0x7
- #define IOCON_MODE_INACT (0x0 << 3)
- #define IOCON_MODE_PULLDOWN (0x1 << 3)
- #define IOCON_MODE_PULLUP (0x2 << 3)
- #define IOCON_MODE_REPEATER (0x3 << 3)
- #define IOCON_HYS_EN (0x1 << 5)
- #define IOCON_GPIO_MODE (0x1 << 5)
- #define IOCON_I2C_SLEW (0x1 << 5)
- #define IOCON_INV_EN (0x1 << 6)
- #define IOCON_ANALOG_EN (0x0 << 7)
- #define IOCON_DIGITAL_EN (0x1 << 7)
- #define IOCON_STDI2C_EN (0x1 << 8)
- #define IOCON_FASTI2C_EN (0x3 << 8)
- #define IOCON_INPFILT_OFF (0x1 << 8)
- #define IOCON_INPFILT_ON (0x0 << 8)
- #define IOCON_OPENDRAIN_EN (0x1 << 10)
- #define IOCON_S_MODE_0CLK (0x0 << 11)
- #define IOCON_S_MODE_1CLK (0x1 << 11)
- #define IOCON_S_MODE_2CLK (0x2 << 11)
- #define IOCON_S_MODE_3CLK (0x3 << 11)
- #define IOCON_S_MODE(clks) ((clks) << 11)
- #define IOCON_CLKDIV(div) ((div) << 13)

### Functions

- __STATIC_INLINE void Chip_IOCON_PinMuxSet (LPC_IOCON_T ∗pIOCON, uint8_t port, uint8_t pin, uint32_t modefunc)

  *Sets I/O Control pin mux.*
- __STATIC_INLINE void Chip_IOCON_PinMux (LPC_IOCON_T ∗pIOCON, uint8_t port, uint8_t pin, uint16↩_t mode, uint8_t func)

  *I/O Control pin mux.*
- __STATIC_INLINE void Chip_IOCON_SetPinMuxing (LPC_IOCON_T ∗pIOCON, const PINMUX_GRP_↩T ∗pinArray, uint32_t arrayLength)

  *Set all I/O Control pin muxing.*

## 8.29 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/lpc_assert.h File Reference

### Macros

- #define LPC_ASSERT(cond, file, line) if(!(cond)){} /∗ Required to avoid unused variable warning ∗/

### 8.29.1 Macro Definition Documentation

#### 8.29.1.1 #define LPC_ASSERT( *cond, file, line* ) if(!(cond)){} /∗ Required to avoid unused variable warning ∗/

Definition at line 54 of file lpc_assert.h.

## 8.30 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/lpc_types.h File Reference

```
#include <stdint.h>
#include <stdbool.h>
```

### Macros

- #define PARAM_SETSTATE(State) ((State == RESET) || (State == SET))
- #define PARAM_FUNCTIONALSTATE(State) ((State == DISABLE) || (State == ENABLE))
- #define _BIT(n) (1 << (n))
- #define _SBF(f, v) ((v) << (f))
- #define _BITMASK(field_width) ( _BIT(field_width) - 1)
- #define NULL ((void ∗) 0)
- #define NELEMENTS(array) (sizeof(array) / sizeof(array[0]))
- #define STATIC static
- #define EXTERN extern
- #define MAX(a, b) (((a) > (b)) ? (a) : (b))
- #define MIN(a, b) (((a) < (b)) ? (a) : (b))
- #define INLINE inline
- #define ALIGN(x) __attribute__ ((aligned(x)))
- #define WEAK __attribute__((weak))

### Typedefs

- typedef enum FlagStatus IntStatus
- typedef enum FlagStatus SetState
- typedef void(∗ PFV) ()
- typedef int32_t(∗ PFI) ()
- typedef char CHAR
- typedef uint8_t UNS_8
- typedef int8_t INT_8
- typedef uint16_t UNS_16
- typedef int16_t INT_16
- typedef uint32_t UNS_32
- typedef int32_t INT_32
- typedef int64_t INT_64
- typedef uint64_t UNS_64
- typedef bool BOOL_32
- typedef bool BOOL_16
- typedef bool BOOL_8

### Enumerations

- enum Bool { FALSE = 0, TRUE = !FALSE }

  *Boolean Type definition.*
- enum FlagStatus { RESET = 0, SET = !RESET }

  *Boolean Type definition.*
- enum FunctionalState { DISABLE = 0, ENABLE = !DISABLE }

  *Functional State Definition.*
- enum Status { ERROR = 0, SUCCESS = !ERROR }
- enum TRANSFER_BLOCK_T { NONE_BLOCKING = 0, BLOCKING }

## 8.31 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/mailbox_5411x.h File Reference

### Data Structures

- struct LPC_MBOXIRQ_T
- struct LPC_MBOX_T

### Macros

- #define MAILBOX_AVAIL (MAILBOX_CM4 + 1) /∗ Number of available mailboxes ∗/

### Enumerations

- enum MBOX_IDX_T { MAILBOX_CM0PLUS = 0, MAILBOX_CM4 }

### Functions

- __STATIC_INLINE void Chip_MBOX_Init (LPC_MBOX_T ∗pMBOX)

  *Initialize mailbox.*
- __STATIC_INLINE void Chip_MBOX_DeInit (LPC_MBOX_T ∗pMBOX)

  *Shutdown mailbox.*
- __STATIC_INLINE void Chip_MBOX_SetValue (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id, uint32_t mboxData)

  *Set data value in the mailbox based on the CPU ID.*
- __STATIC_INLINE void Chip_MBOX_SetValueBits (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id, uint32_t mboxSetBits)

  *Set data bits in the mailbox based on the CPU ID.*
- __STATIC_INLINE void Chip_MBOX_ClearValueBits (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id, uint32_t mboxClrBits)

  *Clear data bits in the mailbox based on the CPU ID.*
- __STATIC_INLINE uint32_t Chip_MBOX_GetValue (LPC_MBOX_T ∗pMBOX, uint32_t cpu_id)

  *Get data in the mailbox based on the cpu_id.*
- __STATIC_INLINE uint32_t Chip_MBOX_GetMutex (LPC_MBOX_T ∗pMBOX)

  *Get MUTEX state and lock mutex.*
- __STATIC_INLINE void Chip_MBOX_SetMutex (LPC_MBOX_T ∗pMBOX)

  *Set MUTEX state.*

## 8.32 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/mrt_5411x.h File Reference

### Data Structures

- struct LPC_MRT_CH_T

  *MRT register block structure.*
- struct LPC_MRT_T

  *MRT register block structure.*

### Macros

- #define MRT_CHANNELS_NUM (4)

  *LPC5411X MRT chip configuration.*
- #define MRT_NO_IDLE_CHANNEL (0x40)
- #define MRT_INTVAL_IVALUE (0xFFFFFF) /∗ Maximum interval load value and mask ∗/

  *MRT register bit fields & masks.*
- #define MRT_INTVAL_LOAD (0x80000000UL) /∗ Force immediate load of timer interval register bit ∗/
- #define MRT_CTRL_INTEN_MASK (0x01)
- #define MRT_CTRL_MODE_MASK (0x06)
- #define MRT_STAT_INTFLAG (0x01)
- #define MRT_STAT_RUNNING (0x02)
- #define LPC_MRT_CH0 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[0])
- #define LPC_MRT_CH1 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[1])
- #define LPC_MRT_CH2 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[2])
- #define LPC_MRT_CH3 ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[3])
- #define LPC_MRT_CH(ch) ((LPC_MRT_CH_T ∗) &LPC_MRT->CHANNEL[(ch)])
- #define MRT0_INTFLAG (1)
- #define MRT1_INTFLAG (2)
- #define MRT2_INTFLAG (4)
- #define MRT3_INTFLAG (8)
- #define MRTn_INTFLAG(ch) (1 << (ch))

## Enumerations

- enum MRT_MODE_T { MRT_MODE_REPEAT = (0 << 1), MRT_MODE_ONESHOT = (1 << 1) }

  *MRT Interrupt Modes enum.*

## Functions

- __STATIC_INLINE void Chip_MRT_Init (void)

  *Initializes the MRT.*
- __STATIC_INLINE void Chip_MRT_DeInit (void)

  *De-initializes the MRT Channel.*
- __STATIC_INLINE LPC_MRT_CH_T ∗ Chip_MRT_GetRegPtr (uint8_t ch)

  *Returns a pointer to the register block for a MRT channel.*
- __STATIC_INLINE uint32_t Chip_MRT_GetInterval (LPC_MRT_CH_T ∗pMRT)

  *Returns the timer time interval value.*
- __STATIC_INLINE void Chip_MRT_SetInterval (LPC_MRT_CH_T ∗pMRT, uint32_t interval)

  *Sets the timer time interval value.*
- __STATIC_INLINE uint32_t Chip_MRT_GetTimer (LPC_MRT_CH_T ∗pMRT)

  *Returns the current timer value.*
- __STATIC_INLINE bool Chip_MRT_GetEnabled (LPC_MRT_CH_T ∗pMRT)

  *Returns true if the timer is enabled.*
- __STATIC_INLINE void Chip_MRT_SetEnabled (LPC_MRT_CH_T ∗pMRT)

  *Enables the timer.*
- __STATIC_INLINE void Chip_MRT_SetDisabled (LPC_MRT_CH_T ∗pMRT)

  *Disables the timer.*
- __STATIC_INLINE MRT_MODE_T Chip_MRT_GetMode (LPC_MRT_CH_T ∗pMRT)

  *Returns the timer mode (repeat or one-shot)*
- __STATIC_INLINE void Chip_MRT_SetMode (LPC_MRT_CH_T ∗pMRT, MRT_MODE_T mode)

  *Sets the timer mode (repeat or one-shot)*
- __STATIC_INLINE bool Chip_MRT_IsRepeatMode (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer is configured in repeat mode.*
- __STATIC_INLINE bool Chip_MRT_IsOneShotMode (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer is configured in one-shot mode.*
- __STATIC_INLINE bool Chip_MRT_IntPending (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer has an interrupt pending.*
- __STATIC_INLINE void Chip_MRT_IntClear (LPC_MRT_CH_T ∗pMRT)

  *Clears the pending interrupt (if any)*
- __STATIC_INLINE bool Chip_MRT_Running (LPC_MRT_CH_T ∗pMRT)

  *Check if the timer is running.*
- __STATIC_INLINE uint8_t Chip_MRT_GetIdleChannel (void)

  *Returns the IDLE channel value.*
- __STATIC_INLINE uint8_t Chip_MRT_GetIdleChannelShifted (void)

  *Returns the IDLE channel value.*
- __STATIC_INLINE uint32_t Chip_MRT_GetIntPending (void)

  *Returns the interrupt pending status for all MRT channels.*
- __STATIC_INLINE bool Chip_MRT_GetIntPendingByChannel (uint8_t ch)

  *Returns the interrupt pending status for a singel MRT channel.*
- __STATIC_INLINE void Chip_MRT_ClearIntPending (uint32_t mask)

  *Clears the interrupt pending status for one or more MRT channels.*

## 8.33 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/packing.h File Reference

### Macros

- #define ALIGNED(n) /∗ Nothing ∗/

### 8.33.1 Macro Definition Documentation

#### 8.33.1.1 #define ALIGNED( *n* ) /∗ Nothing ∗/

Definition at line 37 of file packing.h.

## 8.34 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pinint_5411x.h File Reference

### Data Structures

- struct LPC_PIN_INT_T

  *LPC5411X Pin Interrupt and Pattern Match register block structure.*

### Macros

- #define PININT_ISEL_PMODE_MASK ((uint32_t) 0x00FF)
- #define PININT_PMCTRL_MASK ((uint32_t) 0xFF000003)
- #define PININT_PMCTRL_PMATCH_SEL (1 << 0)
- #define PININT_PMCTRL_RXEV_ENA (1 << 1)
- #define PININT_SRC_BITSOURCE_START 8
- #define PININT_SRC_BITSOURCE_MASK 7
- #define PININT_SRC_BITCFG_START 8
- #define PININT_SRC_BITCFG_MASK 7
- #define PININTCH0 (1 << 0)
- #define PININTCH1 (1 << 1)
- #define PININTCH2 (1 << 2)
- #define PININTCH3 (1 << 3)
- #define PININTCH4 (1 << 4)
- #define PININTCH5 (1 << 5)
- #define PININTCH6 (1 << 6)
- #define PININTCH7 (1 << 7)
- #define PININTCH(ch) (1 << (ch))

### Enumerations

- enum Chip_PININT_SELECT_T {
  PININTSELECT0 = 0, PININTSELECT1 = 1, PININTSELECT2 = 2, PININTSELECT3 = 3,
  PININTSELECT4 = 4, PININTSELECT5 = 5, PININTSELECT6 = 6, PININTSELECT7 = 7 }
- enum Chip_PININT_BITSLICE_T {
  PININTBITSLICE0 = 0, PININTBITSLICE1 = 1, PININTBITSLICE2 = 2, PININTBITSLICE3 = 3,
  PININTBITSLICE4 = 4, PININTBITSLICE5 = 5, PININTBITSLICE6 = 6, PININTBITSLICE7 = 7 }
- enum Chip_PININT_BITSLICE_CFG_T {
  PININT_PATTERNCONST1 = 0x0, PININT_PATTERNRISING = 0x1, PININT_PATTERNFALLING = 0x2,
  PININT_PATTERNRISINGORFALLING = 0x3,
  PININT_PATTERNHIGH = 0x4, PININT_PATTERNLOW = 0x5, PININT_PATTERNCONST0 = 0x6, PININT_PATTERNEVENT = 0x7 }

## Functions

- __STATIC_INLINE void Chip_PININT_Init (LPC_PIN_INT_T ∗pPININT)

  *Initialize Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_DeInit (LPC_PIN_INT_T ∗pPININT)

  *De-Initialize Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPinModeEdge (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Configure the pins as edge sensitive in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPinModeLevel (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Configure the pins as level sensitive in Pin interrupt block.*
- __STATIC_INLINE uint32_t Chip_PININT_GetPinMode (LPC_PIN_INT_T ∗pPININT)

  *Return current PININT edge or level sensitive interrupt selection state.*
- __STATIC_INLINE uint32_t Chip_PININT_GetHighEnabled (LPC_PIN_INT_T ∗pPININT)

  *Return current PININT rising edge or level interrupt enable state.*
- __STATIC_INLINE void Chip_PININT_EnableIntHigh (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Enable rising edge/level PININT interrupts for pins.*
- __STATIC_INLINE void Chip_PININT_DisableIntHigh (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Disable rising edge/level PININT interrupts for pins.*
- __STATIC_INLINE uint32_t Chip_PININT_GetLowEnabled (LPC_PIN_INT_T ∗pPININT)

  *Return current PININT falling edge or level interrupt active level enable state.*
- __STATIC_INLINE void Chip_PININT_EnableIntLow (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Enable falling edge/level active level PININT interrupts for pins.*
- __STATIC_INLINE void Chip_PININT_DisableIntLow (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Disable low edge/level active level PININT interrupts for pins.*
- __STATIC_INLINE uint32_t Chip_PININT_GetRiseStates (LPC_PIN_INT_T ∗pPININT)

  *Return pin states that have a detected latched rising edge (RISE) state.*
- __STATIC_INLINE void Chip_PININT_ClearRiseStates (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Clears pin states that had a latched rising edge (RISE) state.*
- __STATIC_INLINE uint32_t Chip_PININT_GetFallStates (LPC_PIN_INT_T ∗pPININT)

  *Return pin states that have a detected latched falling edge (FALL) state.*
- __STATIC_INLINE void Chip_PININT_ClearFallStates (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Clears pin states that had a latched falling edge (FALL) state.*
- __STATIC_INLINE uint32_t Chip_PININT_GetIntStatus (LPC_PIN_INT_T ∗pPININT)

  *Get interrupt status from Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_ClearIntStatus (LPC_PIN_INT_T ∗pPININT, uint32_t pins)

  *Clear interrupt status in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPatternMatchSrc (LPC_PIN_INT_T ∗pPININT, Chip_PININT_SELECT_T channelNum, Chip_PININT_BITSLICE_T sliceNum)

  *Set source for pattern match in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_SetPatternMatchConfig (LPC_PIN_INT_T ∗pPININT, Chip_PININT_BITSLICE_T sliceNum, Chip_PININT_BITSLICE_CFG_T slice_cfg, bool end_point)

  *Configure the pattern matcch in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_EnablePatternMatch (LPC_PIN_INT_T ∗pPININT)

  *Enable pattern match interrupts in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_DisablePatternMatch (LPC_PIN_INT_T ∗pPININT)

  *Disable pattern match interrupts in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_EnablePatternMatchRxEv (LPC_PIN_INT_T ∗pPININT)

  *Enable RXEV output in Pin interrupt block.*
- __STATIC_INLINE void Chip_PININT_DisablePatternMatchRxEv (LPC_PIN_INT_T ∗pPININT)

  *Disable RXEV output in Pin interrupt block.*
- __STATIC_INLINE uint32_t Chip_PININT_GetPatternMatchState (LPC_PIN_INT_T ∗pPININT)

  *Return pattern match state.*

## 8.35 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pintable_5411x.h File Reference

### Data Structures

- struct PINTABLE_T

  *LPC5411X Pin table structure used for enhanced boot block support.*

### Macros

- #define IMAGE_ENH_MARKER_OFF 0x24
- #define IMAGE_SINGLE_ENH_SIG 0xEDDC9494
- #define IMAGE_DUAL_ENH_SIG 0x0FFEB6B6
- #define IMAGE_BOOT_BLOCK_OFF 0x28
- #define IMAGE_ENH_BLOCK_MARKER 0xFEEDA5A5
- #define SETPORTPIN(port, pin) (((port) & 0x7) << 5) | ((pin) & 0x1F)

### Enumerations

- enum IMAGE_T {
  IMG_NORMAL = 0, IMG_ISP_WAIT, IMG_NO_WAIT, IMG_NO_CRC,
  IMG_JUST_BOOT = 0xFF }
- enum IFSEL_T {
  SL_AUTO = 0, SL_I2C0, SL_I2C1, SL_I2C2,
  SL_SPI0, SL_SPI1 }

## 8.36 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pll_5411x.h File Reference

### Data Structures

- struct PLL_CONFIG_T

  *PLL configuration structure This structure can be used to configure the settings for a PLL setup structure. Fill in the desired configuration for the PLL and call the PLL setup function to fill in a PLL setup structure.*
- struct PLL_SETUP_T

  *PLL setup structure This structure can be used to pre-build a PLL setup configuration at run-time and quickly set the PLL to the configuration. It can be populated with the PLL setup function. If powering up or waiting for PLL lock, the PLL input clock source should be configured prior to PLL setup.*

### Macros

- #define PLL_CONFIGFLAG_USEINRATE (1 << 0)

  *PLL configuration structure flags for 'flags' field These flags control how the PLL configuration function sets up the PLL setup structure.*

- #define PLL_CONFIGFLAG_FORCENOFRACT (1 << 2)
- #define PLL_SETUPFLAG_POWERUP (1 << 0)

  *PLL setup structure flags for 'flags' field These flags control how the PLL setup function sets up the PLL.*
- #define PLL_SETUPFLAG_WAITLOCK (1 << 1)
- #define PLL_SETUPFLAG_ADGVOLT (1 << 2)

## Enumerations

- enum CHIP_SYSCON_PLLCLKSRC_T {
SYSCON_PLLCLKSRC_FRO12MHZ = 0, SYSCON_PLLCLKSRC_CLKIN, SYSCON_PLLCLKSRC_WD↩
T, SYSCON_PLLCLKSRC_RTC,
SYSCON_PLLCLKSRC_DISABLED = 7 }
- enum SS_PROGMODFM_T {
SS_MF_512 = (0 << 20), SS_MF_384 = (1 << 20), SS_MF_256 = (2 << 20), SS_MF_128 = (3 << 20),
SS_MF_64 = (4 << 20), SS_MF_32 = (5 << 20), SS_MF_24 = (6 << 20), SS_MF_16 = (7 << 20) }

    *PLL Spread Spectrum (SS) Programmable modulation frequency See (MF) field in the SYSPLLSSCTRL1 register in the UM.*
- enum SS_PROGMODDP_T {
SS_MR_K0 = (0 << 23), SS_MR_K1 = (1 << 23), SS_MR_K1_5 = (2 << 23), SS_MR_K2 = (3 << 23),
SS_MR_K3 = (4 << 23), SS_MR_K4 = (5 << 23), SS_MR_K6 = (6 << 23), SS_MR_K8 = (7 << 23) }

    *PLL Spread Spectrum (SS) Programmable frequency modulation depth See (MR) field in the SYSPLLSSCTRL1 register in the UM.*
- enum SS_MODWVCTRL_T { SS_MC_NOC = (0 << 26), SS_MC_RECC = (2 << 26), SS_MC_MAXC = (3 << 26) }

    *PLL Spread Spectrum (SS) Modulation waveform control See (MC) field in the SYSPLLSSCTRL1 register in the UM. Compensation for low pass filtering of the PLL to get a triangular modulation at the output of the PLL, giving a flat frequency spectrum.*
- enum PLL_ERROR_T {
PLL_ERROR_SUCCESS = 0, PLL_ERROR_OUTPUT_TOO_LOW, PLL_ERROR_OUTPUT_TOO_HIGH,
PLL_ERROR_INPUT_TOO_LOW,
PLL_ERROR_INPUT_TOO_HIGH, PLL_ERROR_OUTSIDE_INTLIMIT }

    *PLL status definitions.*

## Functions

- __STATIC_INLINE void Chip_Clock_SetSystemPLLSource (CHIP_SYSCON_PLLCLKSRC_T src)

    *Set System PLL clock source.*
- uint32_t Chip_Clock_GetSystemPLLInClockRate (void)

    *Return System PLL input clock rate.*
- uint32_t Chip_Clock_GetSystemPLLOutClockRate (bool recompute)

    *Return System PLL output clock rate.*
- void Chip_Clock_SetBypassPLL (bool bypass)

    *Enables and disables PLL bypass mode.*
- __STATIC_INLINE bool Chip_Clock_IsSystemPLLLocked (void)

    *Check if PLL is locked or not.*
- uint32_t Chip_Clock_GetStoredPLLClockRate (void)

    *Get the rate of pll from the stored value.*
- void Chip_Clock_SetStoredPLLClockRate (uint32_t rate)

    *Store the current PLL rate.*
- uint32_t Chip_Clock_GetSystemPLLOutFromSetup (PLL_SETUP_T ∗pSetup)

    *Return System PLL output clock rate from setup structure.*
- PLL_ERROR_T Chip_Clock_SetupPLLData (PLL_CONFIG_T ∗pControl, PLL_SETUP_T ∗pSetup)

    *Set PLL output based on the passed PLL setup data.*
- PLL_ERROR_T Chip_Clock_SetupSystemPLLPrec (PLL_SETUP_T ∗pSetup)

    *Set PLL output from PLL setup structure (precise frequency)*
- PLL_ERROR_T Chip_Clock_SetPLLFreq (const PLL_SETUP_T ∗pSetup)

    *Set PLL output from PLL setup structure (precise frequency)*
- void Chip_Clock_SetupSystemPLL (uint32_t multiply_by, uint32_t input_freq)

    *Set PLL output based on the multiplier and input frequency.*

## 8.37 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/pmu_5411x.h File Reference

### Data Structures

- struct LPC_PMU_T

  *PMU register block structure.*

### Macros

- #define PMU_BOD_RST (1 << 6)
- #define PMU_BOD_INT (1 << 7)

### Enumerations

- enum CHIP_PMU_BODRSTLVL_T {
  PMU_BODRSTLVL_0, PMU_BODRSTLVL_1_50V = PMU_BODRSTLVL_0, PMU_BODRSTLVL_1, PMU↩_BODRSTLVL_1_85V = PMU_BODRSTLVL_1,
  PMU_BODRSTLVL_2, PMU_BODRSTLVL_2_00V = PMU_BODRSTLVL_2, PMU_BODRSTLVL_3, PMU↩_BODRSTLVL_2_30V = PMU_BODRSTLVL_3 }
- enum CHIP_PMU_BODRINTVAL_T {
  PMU_BODINTVAL_LVL0, PMU_BODINTVAL_2_05v = PMU_BODINTVAL_LVL0, PMU_BODINTVAL_LV↩L1, PMU_BODINTVAL_2_45v = PMU_BODINTVAL_LVL1,
  PMU_BODINTVAL_LVL2, PMU_BODINTVAL_2_75v = PMU_BODINTVAL_LVL2, PMU_BODINTVAL_LV↩L3, PMU_BODINTVAL_3_05v = PMU_BODINTVAL_LVL3 }

### Functions

- __STATIC_INLINE void Chip_PMU_SetBODLevels (CHIP_PMU_BODRSTLVL_T rstlvl, CHIP_PMU_BOD↩RINTVAL_T intlvl)

  *Set brown-out detection interrupt and reset levels.*
- __STATIC_INLINE void Chip_PMU_EnableBODReset (void)

  *Enable brown-out detection reset.*
- __STATIC_INLINE void Chip_PMU_DisableBODReset (void)

  *Disable brown-out detection reset.*
- __STATIC_INLINE void Chip_PMU_EnableBODInt (void)

  *Enable brown-out detection interrupt.*
- __STATIC_INLINE void Chip_PMU_DisableBODInt (void)

  *Disable brown-out detection interrupt.*

## 8.38 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/power_lib_5411x.h File Reference

### Macros

- #define LPC5411X_ROMVER_0 (0x1100)
- #define LPC5411X_ROMVER_1 (0x1101)
- #define LPC5411X_ROMVER_2 (0x1102)

### Enumerations

- enum POWER_MODE_T { POWER_SLEEP = 0, POWER_DEEP_SLEEP, POWER_DEEP_POWER_DO↩WN }

### Functions

- void Chip_POWER_SetFROHFRate (uint32_t freq)

  *Sets the High Frequency FRO rate to (48MHz or 96MHz)*
- uint32_t Chip_POWER_SetPLL (uint32_t multiply_by, uint32_t input_freq)

  *Sets up the System PLL given the PLL input frequency and feedback multiplier.*
- uint32_t Chip_POWER_SetVoltage (uint32_t desired_freq)

  *Set optimal system voltage based on passed system frequency.*
- void Chip_POWER_SetLowPowerVoltage (uint32_t freq)

  *Set low-power voltage levels for LP mode.*
- void Chip_POWER_EnterPowerMode (POWER_MODE_T mode, uint32_t peripheral_ctrl)

  *Enters the selected power state.*
- uint32_t Chip_POWER_GetROMVersion (void)

  *Return ROM version.*

## 8.39 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/ring_buffer.h File Reference

```
#include "lpc_types.h"
#include "cmsis.h"
```

### Data Structures

- struct RINGBUFF_T

  *Ring buffer structure.*

### Macros

- #define RB_VHEAD(rb) (∗(volatile uint32_t ∗) &(rb)->head)
- #define RB_VTAIL(rb) (∗(volatile uint32_t ∗) &(rb)->tail)

### Functions

- int RingBuffer_Init (RINGBUFF_T ∗RingBuff, void ∗buffer, int itemSize, int count, void ∗(∗cpyFunc)(void ∗dst, const void ∗src, uint32_t len))

  *Initialize ring buffer.*
- __STATIC_INLINE void RingBuffer_Flush (RINGBUFF_T ∗RingBuff)

  *Resets the ring buffer to empty.*
- __STATIC_INLINE int RingBuffer_GetSize (RINGBUFF_T ∗RingBuff)

  *Return size the ring buffer.*
- __STATIC_INLINE int RingBuffer_GetCount (RINGBUFF_T ∗RingBuff)

  *Return number of items in the ring buffer.*
- __STATIC_INLINE int RingBuffer_GetFree (RINGBUFF_T ∗RingBuff)

  *Return number of free items in the ring buffer.*
- __STATIC_INLINE int RingBuffer_IsFull (RINGBUFF_T ∗RingBuff)

  *Return number of items in the ring buffer.*
- __STATIC_INLINE int RingBuffer_IsEmpty (RINGBUFF_T ∗RingBuff)

  *Return empty status of ring buffer.*
- int RingBuffer_Insert (RINGBUFF_T ∗RingBuff, const void ∗data)

  *Insert a single item into ring buffer.*

- int RingBuffer_InsertMult (RINGBUFF_T ∗RingBuff, const void ∗data, int num)

    *Insert an array of items into ring buffer.*
- int RingBuffer_Pop (RINGBUFF_T ∗RingBuff, void ∗data)

    *Pop an item from the ring buffer.*
- int RingBuffer_PopMult (RINGBUFF_T ∗RingBuff, void ∗data, int num)

    *Pop an array of items from the ring buffer.*

## 8.40 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/romapi_5411x.h File Reference

```
#include <stdint.h>
#include "iap.h"
#include "error.h"
#include "cmsis.h"
```

### Data Structures

- struct LPC_ROM_API_T

    *High level ROM API structure.*

### Macros

- #define LPC_ROM_API_BASE_LOC 0x03000200UL
- #define LPC_ROM_API (∗(LPC_ROM_API_T ∗ ∗) LPC_ROM_API_BASE_LOC)
- #define IAP_ENTRY_LOCATION 0x03000205

### Functions

- static INLINE void iap_entry (unsigned int cmd_param[5], unsigned int status_result[4])

    *LPC5410x IAP_ENTRY API function type.*

## 8.41 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/rtc_5411x.h File Reference

### Data Structures

- struct LPC_RTC_T

    *LPC5411X Real Time clock register block structure.*

### Macros

- #define RTC_CTRL_SWRESET (1 << 0)
- #define RTC_CTRL_ALARM1HZ (1 << 2)
- #define RTC_CTRL_WAKE1KHZ (1 << 3)
- #define RTC_CTRL_ALARMDPD_EN (1 << 4)
- #define RTC_CTRL_WAKEDPD_EN (1 << 5)
- #define RTC_CTRL_RTC1KHZ_EN (1 << 6)
- #define RTC_CTRL_RTC_EN (1 << 7)
- #define RTC_CTRL_RTC_OSC_PD (1 << 8)
- #define RTC_CTRL_RTC_OSC_BYPASS (1 << 9)
- #define RTC_CTRL_MASK ((uint32_t) 0x0000003FD)

### Functions

- __STATIC_INLINE void Chip_RTC_Init (LPC_RTC_T ∗pRTC)

  *Initialize the RTC peripheral.*
- __STATIC_INLINE void Chip_RTC_DeInit (LPC_RTC_T ∗pRTC)

  *De-initialize the RTC peripheral.*
- __STATIC_INLINE void Chip_RTC_EnableOptions (LPC_RTC_T ∗pRTC, uint32_t flags)

  *Enable RTC options.*
- __STATIC_INLINE void Chip_RTC_DisableOptions (LPC_RTC_T ∗pRTC, uint32_t flags)

  *Disable RTC options.*
- __STATIC_INLINE void Chip_RTC_Reset (LPC_RTC_T ∗pRTC)

  *Reset RTC.*
- __STATIC_INLINE void Chip_RTC_Enable (LPC_RTC_T ∗pRTC)

  *Enables the RTC.*
- __STATIC_INLINE void Chip_RTC_Disable (LPC_RTC_T ∗pRTC)

  *Disables the RTC.*
- __STATIC_INLINE void Chip_RTC_PowerUp (LPC_RTC_T ∗pRTC)

  *Power up the RTC.*
- __STATIC_INLINE void Chip_RTC_PowerDown (LPC_RTC_T ∗pRTC)

  *Disables the RTC.*
- __STATIC_INLINE void Chip_RTC_Enable1KHZ (LPC_RTC_T ∗pRTC)

  *Enables the RTC 1KHz high resolution timer.*
- __STATIC_INLINE void Chip_RTC_Disable1KHZ (LPC_RTC_T ∗pRTC)

  *Disables the RTC 1KHz high resolution timer.*
- __STATIC_INLINE void Chip_RTC_EnableWakeup (LPC_RTC_T ∗pRTC, uint32_t ints)

  *Enables selected RTC wakeup events.*
- __STATIC_INLINE void Chip_RTC_DisableWakeup (LPC_RTC_T ∗pRTC, uint32_t ints)

  *Disables selected RTC wakeup events.*
- __STATIC_INLINE void Chip_RTC_ClearStatus (LPC_RTC_T ∗pRTC, uint32_t stsMask)

  *Clears latched RTC statuses.*
- __STATIC_INLINE uint32_t Chip_RTC_GetStatus (LPC_RTC_T ∗pRTC)

  *Return RTC control/status register.*
- __STATIC_INLINE void Chip_RTC_SetAlarm (LPC_RTC_T ∗pRTC, uint32_t count)

  *Set RTC match value for alarm status/interrupt.*
- __STATIC_INLINE uint32_t Chip_RTC_GetAlarm (LPC_RTC_T ∗pRTC)

  *Return the RTC match value used for alarm status/interrupt.*
- __STATIC_INLINE void Chip_RTC_SetCount (LPC_RTC_T ∗pRTC, uint32_t count)

  *Set RTC match count for 1 second timer count.*
- __STATIC_INLINE uint32_t Chip_RTC_GetCount (LPC_RTC_T ∗pRTC)

  *Get current RTC 1 second timer count.*
- __STATIC_INLINE void Chip_RTC_SetWake (LPC_RTC_T ∗pRTC, uint16_t count)

  *Set RTC wake count countdown value (in mS ticks)*
- __STATIC_INLINE uint16_t Chip_RTC_GetWake (LPC_RTC_T ∗pRTC)

  *Get RTC wake count countdown value.*

## 8.42 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/rtc_ut.h File Reference

```
#include "chip.h"
#include <stdlib.h>
#include <time.h>
```

### Macros

- #define TM_YEAR_BASE (1970)
- #define TM_DAYOFWEEK (4)

### Functions

- void ConvertRtcTime (uint32_t rtcTick, struct tm ∗pTime)

  *Converts a RTC tick time to Universal time.*
- void ConvertTimeRtc (struct tm ∗pTime, uint32_t ∗rtcTick)

  *Converts a Universal time to RTC tick time.*

## 8.43 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/sct_5411x.h File Reference

### Data Structures

- struct LPC_SCT_T

  *State Configurable Timer register block structure.*

### Macros

- #define CONFIG_SCT_nEV (13)
- #define CONFIG_SCT_nRG (13)
- #define CONFIG_SCT_nOU (8)
- #define CONFIG_SCT_nIN (8)
- #define SCT_CONFIG_16BIT_COUNTER 0x00000000

  *Macro defines for SCT configuration register.*
- #define SCT_CONFIG_32BIT_COUNTER 0x00000001
- #define SCT_CONFIG_CLKMODE_BUSCLK (0x0 << 1)
- #define SCT_CONFIG_CLKMODE_SCTCLK (0x1 << 1)
- #define SCT_CONFIG_CLKMODE_INCLK (0x2 << 1)
- #define SCT_CONFIG_CLKMODE_INEDGECLK (0x3 << 1)
- #define SCT_CONFIG_CLKMODE_SYSCLK (0x0 << 1)
- #define SCT_CONFIG_CLKMODE_PRESCALED_SYSCLK (0x1 << 1)
- #define SCT_CONFIG_CLKMODE_SCT_INPUT (0x2 << 1)
- #define SCT_CONFIG_CLKMODE_PRESCALED_SCT_INPUT (0x3 << 1)
- #define SCT_CONFIG_CKSEL_RISING_IN_0 (0x0UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_0 (0x1UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_1 (0x2UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_1 (0x3UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_2 (0x4UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_2 (0x5UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_3 (0x6UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_3 (0x7UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_4 (0x8UL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_4 (0x9UL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_5 (0xAUL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_5 (0xBUL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_6 (0xCUL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_6 (0xDUL << 3)
- #define SCT_CONFIG_CKSEL_RISING_IN_7 (0xEUL << 3)
- #define SCT_CONFIG_CKSEL_FALLING_IN_7 (0xFUL << 3)

- #define SCT_CONFIG_NORELOADL_U (0x1 << 7)
- #define SCT_CONFIG_NORELOADH (0x1 << 8)
- #define SCT_CONFIG_AUTOLIMIT_U (0x1UL << 17)
- #define SCT_CONFIG_AUTOLIMIT_L (0x1UL << 17)
- #define SCT_CONFIG_AUTOLIMIT_H (0x1UL << 18)
- #define COUNTUP_TO_LIMIT_THEN_CLEAR_TO_ZERO 0

  *Macro defines for SCT control register.*
- #define COUNTUP_TO LIMIT_THEN_COUNTDOWN_TO_ZERO 1
- #define SCT_CTRL_STOP_L (1 << 1)
- #define SCT_CTRL_HALT_L (1 << 2)
- #define SCT_CTRL_CLRCTR_L (1 << 3)
- #define SCT_CTRL_BIDIR_L(x) (((x) & 0x01) << 4)
- #define SCT_CTRL_PRE_L(x) (((x) & 0xFF) << 5)
- #define COUNTUP_TO_LIMIT_THEN_CLEAR_TO_ZERO 0

  *Macro defines for SCT control register.*
- #define COUNTUP_TO LIMIT_THEN_COUNTDOWN_TO_ZERO 1
- #define SCT_CTRL_STOP_H (1 << 17)
- #define SCT_CTRL_HALT_H (1 << 18)
- #define SCT_CTRL_CLRCTR_H (1 << 19)
- #define SCT_CTRL_BIDIR_H(x) (((x) & 0x01) << 20)
- #define SCT_CTRL_PRE_H(x) (((x) & 0xFF) << 21)
- #define SCT_EV_CTRL_MATCHSEL(reg) (reg << 0)
- #define SCT_EV_CTRL_HEVENT_L (0UL << 4)
- #define SCT_EV_CTRL_HEVENT_H (1UL << 4)
- #define SCT_EV_CTRL_OUTSEL_INPUT (0UL << 5)
- #define SCT_EV_CTRL_OUTSEL_OUTPUT (0UL << 5)
- #define SCT_EV_CTRL_IOSEL(signal) (signal << 6)
- #define SCT_EV_CTRL_IOCOND_LOW (0UL << 10)
- #define SCT_EV_CTRL_IOCOND_RISE (0x1UL << 10)
- #define SCT_EV_CTRL_IOCOND_FALL (0x2UL << 10)
- #define SCT_EV_CTRL_IOCOND_HIGH (0x3UL << 10)
- #define SCT_EV_CTRL_COMBMODE_OR (0x0UL << 12)
- #define SCT_EV_CTRL_COMBMODE_MATCH (0x1UL << 12)
- #define SCT_EV_CTRL_COMBMODE_IO (0x2UL << 12)
- #define SCT_EV_CTRL_COMBMODE_AND (0x3UL << 12)
- #define SCT_EV_CTRL_STATELD (0x1UL << 14)
- #define SCT_EV_CTRL_STATEV(x) (x << 15)
- #define SCT_EV_CTRL_MATCHMEM (0x1UL << 20)
- #define SCT_EV_CTRL_DIRECTION_INDEPENDENT (0x0UL << 21)
- #define SCT_EV_CTRL_DIRECTION_UP (0x1UL << 21)
- #define SCT_EV_CTRL_DIRECTION_DOWN (0x2UL << 21)
- #define SCT_RES_NOCHANGE (0)

  *Macro defines for SCT Conflict resolution register.*
- #define SCT_RES_SET_OUTPUT (1)
- #define SCT_RES_CLEAR_OUTPUT (2)
- #define SCT_RES_TOGGLE_OUTPUT (3)

## Enumerations

- enum CHIP_SCT_MATCH_REG_T {
SCT_MATCH_0 = 0, SCT_MATCH_1, SCT_MATCH_2, SCT_MATCH_3,
SCT_MATCH_4, SCT_MATCH_5, SCT_MATCH_6, SCT_MATCH_7,
SCT_MATCH_8, SCT_MATCH_9, SCT_MATCH_10, SCT_MATCH_11,
SCT_MATCH_12, SCT_MATCH_13, SCT_MATCH_14, SCT_MATCH_15 }
- enum CHIP_SCT_EVENT_T {
SCT_EVT_0 = (1 << 0), SCT_EVT_1 = (1 << 1), SCT_EVT_2 = (1 << 2), SCT_EVT_3 = (1 << 3),
SCT_EVT_4 = (1 << 4), SCT_EVT_5 = (1 << 5), SCT_EVT_6 = (1 << 6), SCT_EVT_7 = (1 << 7),
SCT_EVT_8 = (1 << 8), SCT_EVT_9 = (1 << 9), SCT_EVT_10 = (1 << 10), SCT_EVT_11 = (1 << 11),
SCT_EVT_12 = (1 << 12), SCT_EVT_13 = (1 << 13), SCT_EVT_14 = (1 << 14), SCT_EVT_15 = (1 << 15) }

## Functions

- __STATIC_INLINE void Chip_SCT_EventControl (LPC_SCT_T ∗pSCT, uint32_t event_number, uint32_t value)

  *Set event control register.*
- __STATIC_INLINE void Chip_SCT_EventStateMask (LPC_SCT_T ∗pSCT, uint32_t event_number, uint32_t event_state_mask)

  *Set event state mask register.*
- __STATIC_INLINE void Chip_SCT_Config (LPC_SCT_T ∗pSCT, uint32_t cfg)

  *Set configuration register.*
- __STATIC_INLINE void Chip_SCT_Limit (LPC_SCT_T ∗pSCT, uint32_t value)

  *Configures the Limit register.*
- void Chip_SCT_SetClrControl (LPC_SCT_T ∗pSCT, uint32_t value, FunctionalState ena)

  *Set or Clear the Control register.*
- void Chip_SCT_SetConflictResolution (LPC_SCT_T ∗pSCT, uint8_t outnum, uint8_t value)

  *Set the conflict resolution.*
- __STATIC_INLINE void Chip_SCT_SetCount (LPC_SCT_T ∗pSCT, uint32_t count)

  *Set unified count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetCountL (LPC_SCT_T ∗pSCT, uint16_t count)

  *Set lower count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetCountH (LPC_SCT_T ∗pSCT, uint16_t count)

  *Set higher count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetMatchCount (LPC_SCT_T ∗pSCT, CHIP_SCT_MATCH_REG_T n, uint32_t value)

  *Set unified match count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetMatchReload (LPC_SCT_T ∗pSCT, CHIP_SCT_MATCH_REG_T n, uint32_t value)

  *Set unified match reload count value in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_EnableEventInt (LPC_SCT_T ∗pSCT, CHIP_SCT_EVENT_T evt)

  *Enable the interrupt for the specified event in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_DisableEventInt (LPC_SCT_T ∗pSCT, CHIP_SCT_EVENT_T evt)

  *Disable the interrupt for the specified event in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_ClearEventFlag (LPC_SCT_T ∗pSCT, CHIP_SCT_EVENT_T evt)

  *Clear the specified event flag in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_SetControl (LPC_SCT_T ∗pSCT, uint32_t value)

  *Set control register in State Configurable Timer.*
- __STATIC_INLINE void Chip_SCT_ClearControl (LPC_SCT_T ∗pSCT, uint32_t value)

  *Clear control register in State Configurable Timer.*
- void Chip_SCT_Init (LPC_SCT_T ∗pSCT)

*Initializes the State Configurable Timer.*
- void Chip_SCT_DeInit (LPC_SCT_T ∗pSCT)

  *Deinitializes the State Configurable Timer.*

## 8.44 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/sct_pwm_5411x.h File Reference

### Functions

- __STATIC_INLINE uint32_t Chip_SCTPWM_GetTicksPerCycle (LPC_SCT_T ∗pSCT)

  *Get number of ticks per PWM cycle.*
- __STATIC_INLINE uint32_t Chip_SCTPWM_PercentageToTicks (LPC_SCT_T ∗pSCT, uint8_t percent)

  *Converts a percentage to ticks.*
- __STATIC_INLINE uint32_t Chip_SCTPWM_GetDutyCycle (LPC_SCT_T ∗pSCT, uint8_t index)

  *Get number of ticks on per PWM cycle.*
- __STATIC_INLINE void Chip_SCTPWM_SetDutyCycle (LPC_SCT_T ∗pSCT, uint8_t index, uint32_t ticks)

  *Get number of ticks on per PWM cycle.*
- __STATIC_INLINE void Chip_SCTPWM_Init (LPC_SCT_T ∗pSCT)

  *Initialize the SCT/PWM clock and reset.*
- __STATIC_INLINE void Chip_SCTPWM_Start (LPC_SCT_T ∗pSCT)

  *Start the SCT PWM.*
- __STATIC_INLINE void Chip_SCTPWM_Stop (LPC_SCT_T ∗pSCT)

  *Stop the SCT PWM.*
- void Chip_SCTPWM_SetRate (LPC_SCT_T ∗pSCT, uint32_t freq)

  *Sets the frequency of the generated PWM wave.*
- void Chip_SCTPWM_SetOutPin (LPC_SCT_T ∗pSCT, uint8_t index, uint8_t pin)

  *Setup the OUTPUT pin and associate it with an index.*

## 8.45 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spi_common_5411x.h File Reference

### Data Structures

- struct LPC_SPI_T

  *SPI register block structure.*
- struct SPI_CFGSETUP_T

### Macros

- #define SPI_CFG_BITMASK (0x0FBD) /∗∗ SPI register bit mask ∗/
- #define SPI_CFG_SPI_EN (1 << 0) /∗∗ SPI Slave Mode Select ∗/
- #define SPI_CFG_SLAVE_EN (0 << 0) /∗∗ SPI Master Mode Select ∗/
- #define SPI_CFG_MASTER_EN (1 << 2) /∗∗ SPI MSB First mode enable ∗/
- #define SPI_CFG_MSB_FIRST_EN (0 << 3) /∗∗ SPI LSB First mode enable ∗/
- #define SPI_CFG_LSB_FIRST_EN (1 << 3) /∗∗ SPI Clock Phase Select ∗/
- #define SPI_CFG_CPHA_FIRST (0 << 4) /∗∗ Capture data on the first edge, Change data on the following edge ∗/
- #define SPI_CFG_CPHA_SECOND (1 << 4) /∗∗ SPI Clock Polarity Select ∗/
- #define SPI_CFG_CPOL_LO (0 << 5) /∗∗ The rest state of the clock (between frames) is low. ∗/
- #define SPI_CFG_CPOL_HI (1 << 5) /∗∗ The rest state of the clock (between frames) is high. ∗/
- #define SPI_CFG_LBM_EN (1 << 7) /∗∗ SPI control 1 loopback mode enable ∗/
- #define SPI_CFG_SPOL_LO (0 << 8) /∗∗ SPI SSEL0 Polarity Select ∗/

- #define SPI_CFG_SPOL_HI (1 << 8) /∗∗ SSEL0 is active High ∗/
- #define SPI_CFG_SPOLNUM_HI(n) (1 << ((n) + 8)) /∗∗ SSELN is active High, selects 0 - 3 ∗/
- #define SPI_DLY_BITMASK (0xFFFF) /∗∗ SPI DLY Register Mask ∗/
- #define SPI_DLY_PRE_DELAY(n) (((n) & 0x0F) << 0) /∗∗ Time in SPI clocks between SSEL assertion and the beginning of a data frame ∗/
- #define SPI_DLY_POST_DELAY(n) (((n) & 0x0F) << 4) /∗∗ Time in SPI clocks between the end of a data frame and SSEL deassertion. ∗/
- #define SPI_DLY_FRAME_DELAY(n) (((n) & 0x0F) << 8) /∗∗ Minimum time in SPI clocks between adjacent data frames. ∗/
- #define SPI_DLY_TRANSFER_DELAY(n) (((n) & 0x0F) << 12) /∗∗ Minimum time in SPI clocks that the SSEL is deasserted between transfers. ∗/
- #define SPI_STAT_BITMASK (0x01F0) /∗∗ SPI STAT Register BitMask ∗/
- #define SPI_STAT_SSA (1 << 4) /∗∗ Slave Select Assert ∗/
- #define SPI_STAT_SSD (1 << 5) /∗∗ Slave Select Deassert ∗/
- #define SPI_STAT_STALLED (1 << 6) /∗∗ Stalled status flag ∗/
- #define SPI_STAT_EOT (1 << 7) /∗∗ End Transfer flag ∗/
- #define SPI_STAT_MSTIDLE (1 << 8) /∗∗ Idle status flag ∗/
- #define SPI_INT_BITMASK (0x0130) /∗∗ SPI interrupt Enable/Disable bits ∗/
- #define SPI_INT_SSAEN (1 << 4) /∗∗ Slave Select is asserted interrupt [BIT-4 of INTENSET/INTENCL↩R/INTSTAT register] ∗/
- #define SPI_INT_SSDEN (1 << 5) /∗∗ Slave Select is deasserted interrupt [BIT-5 of INTENSET/INTENC↩LR/INTSTAT register] ∗/
- #define SPI_INT_MSTIDLE (1 << 8) /∗∗ SPI master is Idle [BIT-8 of INTENSET/INTENCLR/INTSTA↩T register] ∗/
- #define SPI_FIFOCFG_ENABLETX (1 << 0)

  *SPI FIFO Configuration register bits.*
- #define SPI_FIFOCFG_ENABLERX (1 << 1)
- #define SPI_FIFOCFG_DMATX (1 << 12)
- #define SPI_FIFOCFG_DMARX (1 << 13)
- #define SPI_FIFOCFG_WAKETX (1 << 14)
- #define SPI_FIFOCFG_WAKERX (1 << 15)
- #define SPI_FIFOCFG_EMPTYTX (1 << 16)
- #define SPI_FIFOCFG_EMPTYRX (1 << 17)
- #define SPI_FIFO_DEPTH (8) /∗∗ SPI-FIFO How many entries are in the FIFO ∗/

  *Macro defines for FIFO Status register.*
- #define SPI_FIFOSTAT_BITMASK (0x1F1FFB) /∗∗ SPI-FIFO STAT Register BitMask ∗/
- #define SPI_FIFOSTAT_TXERR (1 << 0) /∗∗ SPI-FIFO transmit error ∗/
- #define SPI_FIFOSTAT_RXERR (1 << 1) /∗∗ SPI-FIFO receive error ∗/
- #define SPI_FIFOSTAT_PERINT (1 << 3) /∗∗ SPI-FIFO peripheral (SPI) interrupt ∗/
- #define SPI_FIFOSTAT_TXEMPTY (1 << 4) /∗∗ SPI-FIFO transmitter empty ∗/
- #define SPI_FIFOSTAT_TXNOTFULL (1 << 5) /∗∗ SPI-FIFO transmitter not full ∗/
- #define SPI_FIFOSTAT_RXNOTEMPTY (1 << 6) /∗∗ SPI-FIFO receiver not empty ∗/
- #define SPI_FIFOSTAT_RXFULL (1 << 7) /∗∗ SPI-FIFO receiver not full ∗/
- #define SPI_FIFOSTAT_TXLVL(val) (((val) >> 8) & 0x1F) /∗∗ SPI-FIFO extract transmit level ∗/
- #define SPI_FIFOSTAT_RXLVL(val) (((val) >> 16) & 0x1F) /∗∗ SPI-FIFO extract receive level ∗/
- #define SPI_FIFOTRIG_BITMASK (0x000f0f03) /∗∗ SPI FIFO trigger settings Register BitMask ∗/

  *UART FIFO trigger settings register defines.*
- #define SPI_FIFOTRIG_TXLVLENA (1 << 0)
- #define SPI_FIFOTRIG_RXLVLENA (1 << 1)
- #define SPI_FIFOTRIG_TXLVL(lvl) ((lvl & 0x0f) << 8)
- #define SPI_FIFOTRIG_RXLVL(lvl) ((lvl & 0x0f) << 16)
- #define SPI_FIFOTRIG_TXLVL_DEFAULT 4
- #define SPI_FIFOTRIG_RXLVL_DEFAULT 0
- #define SPI_FIFOINT_BITMASK (0x001F) /∗∗ FIFO interrupt Bit mask ∗/

*Macro defines for SPI Interrupt enable/disable/status [INTSET/INTCLR/INTSTAT and FIFOINTSET/FIFOINTCLR/↩ FIFOINTSTAT registers].*

- #define SPI_FIFOINT_TXERR (1 << 0) /** TX error interrupt [BIT-0 of FIFOINTENSET/FIFOINTENCL↩ R/FIFOINTSTAT register] */
- #define SPI_FIFOINT_RXERR (1 << 1) /** RX error interrupt [BIT-1 of FIFOINTENSET/FIFOINTENCL↩ R/FIFOINTSTAT register] */
- #define SPI_FIFOINT_TXLVL (1 << 2) /** TX FIFO ready interrupt [BIT-2 of FIFOINTENSET/FIFOINTE↩ NCLR/FIFOINTSTAT register] */
- #define SPI_FIFOINT_RXLVL (1 << 3) /** RX Data ready interrupt [BIT-3 of FIFOINTENSET/FIFOINTE↩ NCLR/FIFOINTSTAT register] */
- #define SPI_FIFOINT_PERINT (1 << 4) /** SPI peripheral interrupt [BIT-4 of FIFOINTSTAT register] */
- #define SPI_RXDAT_BITMASK (0x1FFFFF) /** SPI RXDAT Register BitMask */
- #define SPI_RXDAT_DATA(n) ((n) & 0xFFFF) /** Receiver Data */
- #define SPI_RXDAT_RXSSELN(n) ((~((n >> 16) & 0x0f)) & 0x0f) /** Determine the active SSEL pin */
- #define SPI_RXDAT_RXSSELN_ACTIVE (0 << 16) /** The state of SSEL pin is active */
- #define SPI_RXDAT_SOT (1 << 20) /** Start of Transfer flag */
- #define SPI_TXDAT_BITMASK (0xF7FFFFF) /** SPI TXDATCTL Register BitMask */
- #define SPI_TXDAT_DATA(n) ((n) & 0xFFFF) /** SPI Transmit Data */
- #define SPI_TXDAT_CTRLMASK (0xF7F) /** SPI TXDATCTL Register BitMask for control bits only */
- #define SPI_TXDAT_ASSERT_SSEL (0) /** Assert SSEL0 pin */
- #define SPI_TXDAT_ASSERTNUM_SSEL(n) ((~(1 << (n))) & 0x0f) /** Assert SSELN pin */
- #define SPI_TXDAT_DEASSERT_SSEL (1) /** Deassert SSEL0 pin */
- #define SPI_TXDAT_DEASSERTNUM_SSEL(n) (1 << (n)) /** Deassert SSELN pin */
- #define SPI_TXDAT_DEASSERT_ALL (0xF) /** Deassert all SSEL pins */
- #define SPI_TXDAT_EOT (1 << 4) /** End of Transfer flag (TRANSFER_DELAY is applied after sending the current frame) */
- #define SPI_TXDAT_EOF (1 << 5) /** End of Frame flag (FRAME_DELAY is applied after sending the current part) */
- #define SPI_TXDAT_RXIGNORE (1 << 6) /** Receive Ignore Flag */
- #define SPI_TXDAT_FLEN(n) (((n) & 0x0F) << 8) /** Frame length - 1 */
- #define SPI_TXDAT_FLENMASK (0xF << 8) /** Frame length mask */
- #define SPI_TXDAT_DATA(n) ((n) & 0xFFFF) /** SPI Transmit Data */
- #define SPI_DIV_VAL(n) ((n) & 0xFFFF) /** Rate divider value mask (In Master Mode only)*/
- #define Chip_SPI_ReadFIFO Chip_SPI_ReadRawRXFifo
- #define Chip_SPI_ReadFIFOdata Chip_SPI_ReadRXData
- #define Chip_SPI_WriteFIFOcd Chip_SPI_SetTXCTRLData
- #define Chip_SPI_WriteFIFOdata Chip_SPI_WriteTXData
- #define Chip_SPI_FlushFIFOs Chip_SPI_FlushFifos

## Enumerations

- enum ROM_SPI_CLOCK_MODE_T {
  ROM_SPI_CLOCK_CPHA0_CPOL0 = 0, ROM_SPI_CLOCK_MODE0 = ROM_SPI_CLOCK_CPHA0_CP↩ OL0, ROM_SPI_CLOCK_CPHA1_CPOL0 = 1, ROM_SPI_CLOCK_MODE1 = ROM_SPI_CLOCK_CPHA1↩ _CPOL0,
  ROM_SPI_CLOCK_CPHA0_CPOL1 = 2, ROM_SPI_CLOCK_MODE2 = ROM_SPI_CLOCK_CPHA0_CP↩ OL1, ROM_SPI_CLOCK_CPHA1_CPOL1 = 3, ROM_SPI_CLOCK_MODE3 = ROM_SPI_CLOCK_CPHA1↩ _CPOL1 }

  *SPI Clock Mode.*
- enum SPI_CLOCK_MODE_T {
  SPI_CLOCK_CPHA0_CPOL0 = SPI_CFG_CPOL_LO | SPI_CFG_CPHA_FIRST, SPI_CLOCK_MODE0 = SPI_CLOCK_CPHA0_CPOL0, SPI_CLOCK_CPHA1_CPOL0 = SPI_CFG_CPOL_LO | SPI_CFG_CPHA_↩ SECOND, SPI_CLOCK_MODE1 = SPI_CLOCK_CPHA1_CPOL0,
  SPI_CLOCK_CPHA0_CPOL1 = SPI_CFG_CPOL_HI | SPI_CFG_CPHA_FIRST, SPI_CLOCK_MODE2 = SPI_CLOCK_CPHA0_CPOL1, SPI_CLOCK_CPHA1_CPOL1 = SPI_CFG_CPOL_HI | SPI_CFG_CPHA↩ _SECOND, SPI_CLOCK_MODE3 = SPI_CLOCK_CPHA1_CPOL1 }

  *SPI Clock Mode.*

### Functions

- int Chip_SPI_Init (LPC_SPI_T ∗pSPI)

  *Initialize the SPI.*
- __STATIC_INLINE void Chip_SPI_DeInit (LPC_SPI_T ∗pSPI)

  *Disable SPI operation.*
- __STATIC_INLINE void Chip_SPI_SetCFGRegBits (LPC_SPI_T ∗pSPI, uint32_t bits)

  *Set SPI CFG register values.*
- __STATIC_INLINE void Chip_SPI_ClearCFGRegBits (LPC_SPI_T ∗pSPI, uint32_t bits)

  *Clear SPI CFG register values.*
- __STATIC_INLINE void Chip_SPI_Enable (LPC_SPI_T ∗pSPI)

  *Enable SPI peripheral.*
- __STATIC_INLINE void Chip_SPI_Disable (LPC_SPI_T ∗pSPI)

  *Disable SPI peripheral.*
- __STATIC_INLINE void Chip_SPI_EnableSlaveMode (LPC_SPI_T ∗pSPI)

  *Enable SPI slave mode.*
- __STATIC_INLINE void Chip_SPI_EnableLSBFirst (LPC_SPI_T ∗pSPI)

  *Enable LSB First transfers.*
- __STATIC_INLINE void Chip_SPI_EnableMSBFirst (LPC_SPI_T ∗pSPI)

  *Enable MSB First transfers.*
- __STATIC_INLINE void Chip_SPI_SetSPIMode (LPC_SPI_T ∗pSPI, SPI_CLOCK_MODE_T mode)

  *Set SPI mode.*
- __STATIC_INLINE void Chip_SPI_SetCSPolHigh (LPC_SPI_T ∗pSPI, uint8_t csNum)

  *Set polarity on the SPI chip select high.*
- __STATIC_INLINE void Chip_SPI_SetCSPolLow (LPC_SPI_T ∗pSPI, uint8_t csNum)

  *Set polarity on the SPI chip select low.*
- void Chip_SPI_ConfigureSPI (LPC_SPI_T ∗pSPI, SPI_CFGSETUP_T ∗pCFG)

  *Setup SPI configuration.*
- __STATIC_INLINE uint32_t Chip_SPI_GetStatus (LPC_SPI_T ∗pSPI)

  *Get the current status of SPI controller.*
- __STATIC_INLINE void Chip_SPI_ClearStatus (LPC_SPI_T ∗pSPI, uint32_t Flag)

  *Clear SPI status.*
- __STATIC_INLINE void Chip_SPI_EnableInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Enable a SPI interrupt.*
- __STATIC_INLINE void Chip_SPI_DisableInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Disable a SPI interrupt.*
- __STATIC_INLINE uint32_t Chip_SPI_GetEnabledInts (LPC_SPI_T ∗pSPI)

  *Return enabled SPI interrupts.*
- __STATIC_INLINE uint32_t Chip_SPI_GetPendingInts (LPC_SPI_T ∗pSPI)

  *Return pending SPI interrupts.*
- __STATIC_INLINE void Chip_SPI_SetFIFOCfg (LPC_SPI_T ∗pSPI, uint32_t cfg)

  *Set FIFO Configuration register.*
- __STATIC_INLINE void Chip_SPI_ClearFIFOCfg (LPC_SPI_T ∗pSPI, uint32_t cfg)

  *Clear FIFO Configuration register.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOStatus (LPC_SPI_T ∗pSPI)

  *Get the current status of SPI controller FIFO.*
- __STATIC_INLINE void Chip_SPI_ClearFIFOStatus (LPC_SPI_T ∗pSPI, uint32_t mask)

  *Clear the FIFO status register.*
- __STATIC_INLINE void Chip_SPI_SetFIFOTrigLevel (LPC_SPI_T ∗pSPI, uint8_t tx_lvl, uint8_t rx_lvl)

  *Setup SPI FIFO trigger-level.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOTrigLevel (LPC_SPI_T ∗pSPI)

*Get SPI FIFO trigger-level.*

- __STATIC_INLINE void Chip_SPI_EnableFIFOInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Enable a SPI FIFO interrupt.*
- __STATIC_INLINE void Chip_SPI_DisableFIFOInts (LPC_SPI_T ∗pSPI, uint32_t intMask)

  *Disable a SPI FIFO interrupt.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOEnabledInts (LPC_SPI_T ∗pSPI)

  *Return enabled SPI FIFO interrupts.*
- __STATIC_INLINE uint32_t Chip_SPI_GetFIFOPendingInts (LPC_SPI_T ∗pSPI)

  *Return pending SPI FIFO interrupts.*
- __STATIC_INLINE uint32_t Chip_SPI_ReadRawRXFifo (LPC_SPI_T ∗pSPI)

  *Read raw data from receive FIFO with status bits.*
- __STATIC_INLINE uint16_t Chip_SPI_ReadRXData (LPC_SPI_T ∗pSPI)

  *Read data from receive FIFO masking off status bits.*
- __STATIC_INLINE void Chip_SPI_WriteFIFO (LPC_SPI_T ∗pSPI, uint32_t data)

  *Write FIFOWR register: writes 32-bit value to FIFO.*
- __STATIC_INLINE void Chip_SPI_SetTXCTRLData (LPC_SPI_T ∗pSPI, uint16_t ctrl, uint16_t data)

  *Write FIFOWR register: writes control options and data.*
- __STATIC_INLINE void Chip_SPI_WriteTXData (LPC_SPI_T ∗pSPI, uint16_t data)

  *Write data to transmit FIFO.*
- __STATIC_INLINE void Chip_SPI_FlushFifos (LPC_SPI_T ∗pSPI)

  *Flush FIFOs.*

## 8.46 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spim_5411x.h File Reference

```
#include "spi_common_5411x.h"
```

### Data Structures

- struct SPIM_DELAY_CONFIG_T

  *SPI Delay Configure Struct.*
- struct SPIM_XFER_T

  *SPI Master transfer data context.*

### Macros

- #define SPIM_XFER_OPT_FRAME_DLY (1 << 0) /∗ frame delay between frames ∗/
- #define SPIM_XFER_OPT_FRAME_ASSERT (1 << 1) /∗ assert/de-assert ssel for each frame ∗/
- #define SPIM_XFER_OPT_DMA (1 << 2) /∗ use DMA ∗/

### Enumerations

- enum SPIM_EVENT_T {
  SPIM_EVENT_WAIT, SPIM_EVENT_ERRORTX, SPIM_EVENT_ERRORRX, SPIM_EVENT_ERROR,
  SPIM_EVENT_DONE }

  *SPI master Xfer events.*
- enum SPIM_XFER_STATE_T {
  SPI_XFER_STATE_IDLE, SPI_XFER_STATE_BUSY, SPI_XFER_STATE_DONE, SPI_XFER_STATE_STALL,
  SPI_XFER_STATE_ERROR }

  *States of SPI Master Xfer.*

### Functions

- __STATIC_INLINE uint32_t Chip_SPIM_GetClockRate (LPC_SPI_T ∗pSPI)

  *Get SPI master bit rate.*
- uint32_t Chip_SPIM_SetClockRate (LPC_SPI_T ∗pSPI, uint32_t rate)

  *Set SPI master bit rate.*
- void Chip_SPIM_DelayConfig (LPC_SPI_T ∗pSPI, SPIM_DELAY_CONFIG_T ∗pConfig)

  *Config SPI Delay parameters.*
- __STATIC_INLINE void Chip_SPIM_ForceEndOfTransfer (LPC_SPI_T ∗pSPI)

  *Forces an end of transfer for the current master transfer.*
- __STATIC_INLINE void Chip_SPIM_EnableLoopBack (LPC_SPI_T ∗pSPI)

  *Enable loopback mode.*
- __STATIC_INLINE void Chip_SPIM_DisableLoopBack (LPC_SPI_T ∗pSPI)

  *Disable loopback mode.*
- void Chip_SPIM_XferHandler (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *SPI master transfer state change handler.*
- void Chip_SPIM_Xfer (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Start non-blocking SPI master transfer.*
- void Chip_SPIM_XferFIFO (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Start polled SPI master transfer.*
- void Chip_SPIM_XferBlocking (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Perform blocking SPI master transfer.*

## 8.47 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/spis_5411x.h File Reference

```
#include "spi_common_5411x.h"
```

### Data Structures

- struct SPIS_XFER_T

### Enumerations

- enum SPIS_EVENT_T {
  SPIS_EVENT_SASSERT, SPIS_EVENT_SDEASSERT, SPIS_EVENT_DONE, SPIS_EVENT_ERRORTX,
  SPIS_EVENT_ERRORRX, SPIS_EVENT_THRESHOLD }

  *Slave callback events.*

### Functions

- void Chip_SPIS_Init (LPC_SPI_T ∗pSPI)

  *SPI slave initialization.*
- void Chip_SPIS_EnableInts (LPC_SPI_T ∗pSPI)

  *SPI slave interrupt enable.*
- void Chip_SPIS_DisableInts (LPC_SPI_T ∗pSPI)

  *SPI slave interrupt disable.*
- void Chip_SPIS_LoadFIFO (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *Load slave transmit FIFO.*
- void Chip_SPIS_ReadFIFO (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

*SPI slave FIFO read.*

- void Chip_SPIS_XferHandler (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *SPI slave transfer state change handler.*

## 8.48 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/stopwatch.h File Reference

```
#include "cmsis.h"
```

### Functions

- void StopWatch_Init (void)

  *Initialize stopwatch.*
- uint32_t StopWatch_Start (void)

  *Start a stopwatch.*
- __STATIC_INLINE uint32_t StopWatch_Elapsed (uint32_t startTime)

  *Returns number of ticks elapsed since stopwatch was started.*
- uint32_t StopWatch_TicksPerSecond (void)

  *Returns number of ticks per second of the stopwatch timer.*
- uint32_t StopWatch_TicksToMs (uint32_t ticks)

  *Converts from stopwatch ticks to mS.*
- uint32_t StopWatch_TicksToUs (uint32_t ticks)

  *Converts from stopwatch ticks to uS.*
- uint32_t StopWatch_MsToTicks (uint32_t mS)

  *Converts from mS to stopwatch ticks.*
- uint32_t StopWatch_UsToTicks (uint32_t uS)

  *Converts from uS to stopwatch ticks.*
- __STATIC_INLINE void StopWatch_DelayTicks (uint32_t ticks)

  *Delays the given number of ticks using stopwatch primitives.*
- __STATIC_INLINE void StopWatch_DelayMs (uint32_t mS)

  *Delays the given number of mS using stopwatch primitives.*
- __STATIC_INLINE void StopWatch_DelayUs (uint32_t uS)

  *Delays the given number of uS using stopwatch primitives.*

## 8.49 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/syscon_5411x.h File Reference

### Data Structures

- struct LPC_SYSCON_T

  *LPC5411X Main system configuration register block structure.*
- struct LPC_ASYNC_SYSCON_T

  *LPC5411X Asynchronous system configuration register block structure.*

## Macros

- #define SYSCON_FROCTRL_MASK ((1 << 15) | (0xF << 26))

  *FROCTRL register bits.*
- #define SYSCON_FROCTRL_WRTRIM (1UL << 31)
- #define SYSCON_FROCTRL_HSPDCLK (1UL << 30)
- #define SYSCON_FROCTRL_USBMODCHG (1UL << 25)
- #define SYSCON_FROCTRL_USBCLKADJ (1UL << 24)
- #define SYSCON_FROCTRL_SEL96MHZ (1UL << 14)
- #define SYSCON_NMISRC_M0_ENABLE ((uint32_t) 1 << 30)
- #define SYSCON_NMISRC_M4_ENABLE ((uint32_t) 1 << 31)
- #define SYSCON_RST_POR (1 << 0)
- #define SYSCON_RST_EXTRST (1 << 1)
- #define SYSCON_RST_WDT (1 << 2)
- #define SYSCON_RST_BOD (1 << 3)
- #define SYSCON_RST_SYSRST (1 << 4)
- #define SYSCON_PDRUNCFG_PD_FRO (1 << 4)
- #define SYSCON_PDRUNCFG_PD_FLASH (1 << 5)
- #define SYSCON_PDRUNCFG_PD_TS (1 << 6)
- #define SYSCON_PDRUNCFG_PD_BOD_RST (1 << 7)
- #define SYSCON_PDRUNCFG_PD_BOD_INTR (1 << 8)
- #define SYSCON_PDRUNCFG_PD_ADC0 (1 << 10)
- #define SYSCON_PDRUNCFG_PD_VDDFLASH (1 << 11)
- #define SYSCON_PDRUNCFG_LP_VDDFLASH (1 << 12)
- #define SYSCON_PDRUNCFG_PD_SRAM0 (1 << 13)
- #define SYSCON_PDRUNCFG_PD_SRAM1 (1 << 14)
- #define SYSCON_PDRUNCFG_PD_SRAM2 (1 << 15)
- #define SYSCON_PDRUNCFG_PD_SRAMX (1 << 16)
- #define SYSCON_PDRUNCFG_PD_ROM (1 << 17)
- #define SYSCON_PDRUNCFG_PD_VDDHV_ENA (1 << 18)
- #define SYSCON_PDRUNCFG_PD_VDDA_ENA (1 << 19)
- #define SYSCON_PDRUNCFG_PD_WDT_OSC (1 << 20)
- #define SYSCON_PDRUNCFG_PD_USB_PHY (1 << 21)
- #define SYSCON_PDRUNCFG_PD_SYS_PLL (1 << 22)
- #define SYSCON_PDRUNCFG_PD_VREFP (1 << 23)
- #define SYSCON_AUTOCGOR_RAM0X (1 << 1)
- #define SYSCON_AUTOCGOR_RAM1 (1 << 2)
- #define SYSCON_AUTOCGOR_RAM2 (1 << 3)
- #define SYSCON_AUTOCGOR_MASK (SYSCON_AUTOCGOR_RAM0X | SYSCON_AUTOCGOR_RAM1 | SYSCON_AUTOCGOR_RAM2)

## Enumerations

- enum CHIP_SYSCON_BOOT_MODE_REMAP_T { REMAP_BOOT_LOADER_MODE, REMAP_USER_RAM_MODE, REMAP_USER_FLASH_MODE }
- enum CHIP_SYSCON_PERIPH_RESET_T {
  RESET_FLASH = 7, RESET_FMC, RESET_SPIFI = 10, RESET_MUX,
  RESET_IOCON = 13, RESET_GPIO0, RESET_GPIO1, RESET_PINT = 18,
  RESET_GINT, RESET_DMA, RESET_CRC, RESET_WWDT,
  RESET_ADC = 27, RESET_ADC0 = 27, RESET_MRT = 32, RESET_SCT0 = 32 + 2,
  RESET_SCT = 32 + 2, RESET_UTICK = 32 + 10, RESET_FLEXCOMM0, RESET_FLEXCOMM1,
  RESET_FLEXCOMM2, RESET_FLEXCOMM3, RESET_FLEXCOMM4, RESET_FLEXCOMM5,
  RESET_FLEXCOMM6, RESET_FLEXCOMM7, RESET_DMIC, RESET_TIMER2 = 32 + 22,
  RESET_USB, RESET_TIMER0, RESET_TIMER1, RESET_TIMER3 = 128 + 13,
  RESET_TIMER4 }

- enum SYSCON_FLASHTIM_T {
  SYSCON_FLASH_1CYCLE = 0, FLASHTIM_20MHZ_CPU = SYSCON_FLASH_1CYCLE, SYSCON_FLASH_2CYCLE, SYSCON_FLASH_3CYCLE,
  SYSCON_FLASH_4CYCLE, SYSCON_FLASH_5CYCLE, SYSCON_FLASH_6CYCLE, SYSCON_FLASH_7CYCLE,
  SYSCON_FLASH_8CYCLE }

  *FLASH Access time definitions.*
- enum CHIP_SYSCON_WAKEUP_T {
  SYSCON_STARTER_WWDT_BOD = 0, SYSCON_STARTER_DMA, SYSCON_STARTER_GINT0, SYSCON_STARTER_GINT1,
  SYSCON_STARTER_PINT0, SYSCON_STARTER_PINT1, SYSCON_STARTER_PINT2, SYSCON_STARTER_PINT3,
  SYSCON_STARTER_UTICK, SYSCON_STARTER_MRT, SYSCON_STARTER_TIMER0, SYSCON_STARTER_TIMER1,
  SYSCON_STARTER_SCT0, SYSCON_STARTER_TIMER3, SYSCON_STARTER_FLEXCOMM0, SYSCON_STARTER_FLEXCOMM1,
  SYSCON_STARTER_FLEXCOMM2, SYSCON_STARTER_FLEXCOMM3, SYSCON_STARTER_FLEXCOMM4, SYSCON_STARTER_FLEXCOMM5,
  SYSCON_STARTER_FLEXCOMM6, SYSCON_STARTER_FLEXCOMM7, SYSCON_STARTER_ADC0_SEQA, SYSCON_STARTER_ADC0_SEQB,
  SYSCON_STARTER_ADC0_THCMP, SYSCON_STARTER_DMIC, SYSCON_STARTER_HWVAD, SYSCON_STARTER_USBNEEDCLK,
  SYSCON_STARTER_USB, SYSCON_STARTER_RTC, SYSCON_STARTER_RESERVED0, SYSCON_STARTER_MAILBOX,
  SYSCON_STARTER_PINT4, SYSCON_STARTER_PINT5, SYSCON_STARTER_PINT6, SYSCON_STARTER_PINT7,
  SYSCON_STARTER_TIMER2, SYSCON_STARTER_TIMER4 }

## Functions

- __STATIC_INLINE void Chip_SYSCON_Map (CHIP_SYSCON_BOOT_MODE_REMAP_T remap)

  *Re-map interrupt vectors.*
- __STATIC_INLINE CHIP_SYSCON_BOOT_MODE_REMAP_T Chip_SYSCON_GetMemoryMap (void)

  *Get system remap setting.*
- __STATIC_INLINE void Chip_SYSCON_SetSYSTCKCAL (uint32_t sysCalVal)

  *Set System tick timer calibration value.*
- void Chip_SYSCON_SetNMISource (uint32_t intsrc)

  *Set source for non-maskable interrupt (NMI)*
- void Chip_SYSCON_EnableNMISource (void)

  *Enable interrupt used for NMI source.*
- void Chip_SYSCON_DisableNMISource (void)

  *Disable interrupt used for NMI source.*
- void Chip_SYSCON_Enable_ASYNC_Syscon (bool enable)

  *Enable or disable asynchronous APB bridge and subsystem.*
- __STATIC_INLINE void Chip_SYSCON_SetUSARTFRGCtrl (uint8_t fmul, uint8_t fdiv)

  *Set UART Fractional divider value.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetSystemRSTStatus (void)

  *Get system reset status.*
- __STATIC_INLINE void Chip_SYSCON_ClearSystemRSTStatus (uint32_t reset)

  *Clear system reset status.*
- __STATIC_INLINE void Chip_SYSCON_PeriphReset (CHIP_SYSCON_PERIPH_RESET_T periph)

  *Resets a peripheral.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetPORPIOStatus (uint8_t port)

  *Read POR captured PIO status.*

- __STATIC_INLINE uint32_t Chip_SYSCON_GetResetPIOStatus (uint8_t port)

  *Read reset captured PIO status.*
- __STATIC_INLINE void Chip_SYSCON_StartFreqMeas (void)

  *Starts a frequency measurement cycle.*
- __STATIC_INLINE bool Chip_SYSCON_IsFreqMeasComplete (void)

  *Indicates when a frequency measurement cycle is complete.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetRawFreqMeasCapval (void)

  *Returns the raw capture value for a frequency measurement cycle.*
- uint32_t Chip_SYSCON_GetCompFreqMeas (uint32_t refClockRate)

  *Returns the computed value for a frequency measurement cycle.*
- __STATIC_INLINE void Chip_SYSCON_SetFLASHAccess (SYSCON_FLASHTIM_T clks)

  *Set FLASH memory access time in clocks.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetPowerStates (void)

  *Power up one or more blocks or peripherals.*
- __STATIC_INLINE void Chip_SYSCON_PowerDown (uint32_t powerdownmask)

  *Power down one or more blocks or peripherals.*
- void Chip_SYSCON_PowerUp (uint32_t powerupmask)

  *Power up one or more blocks or peripherals.*
- __STATIC_INLINE void Chip_SYSCON_EnableWakeup (CHIP_SYSCON_WAKEUP_T periphId)

  *Enables a pin's (PINT) wakeup logic.*
- __STATIC_INLINE void Chip_SYSCON_DisableWakeup (CHIP_SYSCON_WAKEUP_T periphId)

  *Disables peripheral's wakeup logic.*
- __STATIC_INLINE uint32_t Chip_SYSCON_GetDeviceID (void)

  *Return the pointer to device ID registers.*
- __STATIC_INLINE void Chip_SYSCON_DisableAutoClocking (uint32_t mask)

  *Disables Auto clock gating for SRAM's.*
- __STATIC_INLINE void Chip_SYSCON_EnableAutoClocking (uint32_t mask)

  *Re-enables Auto clock gating for SRAM's.*

## 8.50 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/timer_5411x.h File Reference

### Data Structures

- struct LPC_TIMER_T

  *32-bit Standard timer register block structure*

### Macros

- #define TIMER_IR_CLR(n) _BIT(n)
- #define TIMER_MATCH_INT(n) (_BIT((n) & 0x0F))
- #define TIMER_CAP_INT(n) (_BIT((((n) & 0x0F) + 4)))
- #define TIMER_ENABLE ((uint32_t) (1 << 0))
- #define TIMER_RESET ((uint32_t) (1 << 1))
- #define TIMER_CTRL_MASK ((uint32_t) 0x03)
- #define TIMER_INT_ON_MATCH(n) (_BIT(((n) ∗ 3)))
- #define TIMER_RESET_ON_MATCH(n) (_BIT((((n) ∗ 3) + 1)))
- #define TIMER_STOP_ON_MATCH(n) (_BIT((((n) ∗ 3) + 2)))
- #define TIMER_MCR_MASK ((uint32_t) 0x0FFF)
- #define TIMER_CAP_RISING(n) (_BIT(((n) ∗ 3)))
- #define TIMER_CAP_FALLING(n) (_BIT((((n) ∗ 3) + 1)))

- #define TIMER_INT_ON_CAP(n) (_BIT((((n) ∗ 3) + 2)))
- #define TIMER_CCR_MASK ((uint32_t) 0x0FFF)
- #define TIMER_EMR_MASK ((uint32_t) 0x0FFF)
- #define TIMER_CTCR_MASK ((uint32_t) 0x0F)

## Enumerations

- enum TIMER_PIN_MATCH_STATE_T { TIMER_EXTMATCH_DO_NOTHING = 0, TIMER_EXTMATCH_↩ CLEAR = 1, TIMER_EXTMATCH_SET = 2, TIMER_EXTMATCH_TOGGLE = 3 }

  *Standard timer initial match pin state and change state.*
- enum TIMER_CAP_SRC_STATE_T { TIMER_CAPSRC_RISING_PCLK = 0, TIMER_CAPSRC_RISING_↩ CAPN = 1, TIMER_CAPSRC_FALLING_CAPN = 2, TIMER_CAPSRC_BOTH_CAPN = 3 }

  *Standard timer clock and edge for count source.*

## Functions

- __STATIC_INLINE void Chip_TIMER_Init (LPC_TIMER_T ∗pTMR)

  *Initialize a timer.*
- __STATIC_INLINE void Chip_TIMER_DeInit (LPC_TIMER_T ∗pTMR)

  *Shutdown a timer.*
- __STATIC_INLINE bool Chip_TIMER_MatchPending (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Determine if a match interrupt is pending.*
- __STATIC_INLINE bool Chip_TIMER_CapturePending (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Determine if a capture interrupt is pending.*
- __STATIC_INLINE void Chip_TIMER_ClearMatch (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Clears a (pending) match interrupt.*
- __STATIC_INLINE void Chip_TIMER_ClearCapture (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Clears a (pending) capture interrupt.*
- __STATIC_INLINE void Chip_TIMER_Enable (LPC_TIMER_T ∗pTMR)

  *Enables the timer (starts count)*
- __STATIC_INLINE void Chip_TIMER_Disable (LPC_TIMER_T ∗pTMR)

  *Disables the timer (stops count)*
- __STATIC_INLINE uint32_t Chip_TIMER_ReadCount (LPC_TIMER_T ∗pTMR)

  *Returns the current timer count.*
- __STATIC_INLINE uint32_t Chip_TIMER_ReadPrescale (LPC_TIMER_T ∗pTMR)

  *Returns the current prescale count.*
- __STATIC_INLINE void Chip_TIMER_PrescaleSet (LPC_TIMER_T ∗pTMR, uint32_t prescale)

  *Sets the prescaler value.*
- __STATIC_INLINE void Chip_TIMER_SetMatch (LPC_TIMER_T ∗pTMR, int8_t matchnum, uint32_t matchval)

  *Sets a timer match value.*
- __STATIC_INLINE uint32_t Chip_TIMER_ReadCapture (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Reads a capture register.*
- void Chip_TIMER_Reset (LPC_TIMER_T ∗pTMR)

  *Resets the timer terminal and prescale counts to 0.*
- __STATIC_INLINE void Chip_TIMER_MatchEnableInt (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Enables a match interrupt that fires when the terminal count matches the match counter value.*
- __STATIC_INLINE void Chip_TIMER_MatchDisableInt (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Disables a match interrupt for a match counter.*
- __STATIC_INLINE void Chip_TIMER_ResetOnMatchEnable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *For the specific match counter, enables reset of the terminal count register when a match occurs.*

- __STATIC_INLINE void Chip_TIMER_ResetOnMatchDisable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *For the specific match counter, disables reset of the terminal count register when a match occurs.*
- __STATIC_INLINE void Chip_TIMER_StopOnMatchEnable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Enable a match timer to stop the terminal count when a match count equals the terminal count.*
- __STATIC_INLINE void Chip_TIMER_StopOnMatchDisable (LPC_TIMER_T ∗pTMR, int8_t matchnum)

  *Disable stop on match for a match timer. Disables a match timer to stop the terminal count when a match count equals the terminal count.*
- __STATIC_INLINE void Chip_TIMER_CaptureRisingEdgeEnable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Enables capture on on rising edge of selected CAP signal for the selected capture register, enables the selected CAPn.capnum signal to load the capture register with the terminal coount on a rising edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureRisingEdgeDisable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Disables capture on on rising edge of selected CAP signal. For the selected capture register, disables the selected CAPn.capnum signal to load the capture register with the terminal coount on a rising edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureFallingEdgeEnable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Enables capture on on falling edge of selected CAP signal. For the selected capture register, enables the selected CAPn.capnum signal to load the capture register with the terminal coount on a falling edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureFallingEdgeDisable (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Disables capture on on falling edge of selected CAP signal. For the selected capture register, disables the selected CAPn.capnum signal to load the capture register with the terminal coount on a falling edge.*
- __STATIC_INLINE void Chip_TIMER_CaptureEnableInt (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Enables interrupt on capture of selected CAP signal. For the selected capture register, an interrupt will be generated when the enabled rising or falling edge on CAPn.capnum is detected.*
- __STATIC_INLINE void Chip_TIMER_CaptureDisableInt (LPC_TIMER_T ∗pTMR, int8_t capnum)

  *Disables interrupt on capture of selected CAP signal.*
- void Chip_TIMER_ExtMatchControlSet (LPC_TIMER_T ∗pTMR, int8_t initial_state, TIMER_PIN_MATCH_STATE_T matchState, int8_t matchnum)

  *Sets external match control (MATn.matchnum) pin control. For the pin selected with matchnum, sets the function of the pin that occurs on a terminal count match for the match count.*
- __STATIC_INLINE void Chip_TIMER_TIMER_SetCountClockSrc (LPC_TIMER_T ∗pTMR, TIMER_CAP_SRC_STATE_T capSrc, int8_t capnum)

  *Sets timer count source and edge with the selected passed from CapSrc. If CapSrc selected a CAPn pin, select the specific CAPn pin with the capnum value.*

## 8.51 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/uart_5411x.h File Reference

```
#include "ring_buffer.h"
```

### Data Structures

- struct LPC_USART_T

  *UART Registers.*
- struct UART_BAUD_T

  *UART Baud rate calculation structure.*
- struct UART_STATISTICS_T

  *UART statistics structure.*

## Macros

- #define ECHO_EN 1
- #define ECHO_DIS 0
- #define UART_CFG_BITMASK (0x00fddbfd)

  *UART CFG register definitions.*
- #define UART_CFG_ENABLE (0x01 << 0)
- #define UART_CFG_DATALEN_7 (0x00 << 2)
- #define UART_CFG_DATALEN_8 (0x01 << 2)
- #define UART_CFG_DATALEN_9 (0x02 << 2)
- #define UART_CFG_PARITY_NONE (0x00 << 4)
- #define UART_CFG_PARITY_EVEN (0x02 << 4)
- #define UART_CFG_PARITY_ODD (0x03 << 4)
- #define UART_CFG_STOPLEN_1 (0x00 << 6)
- #define UART_CFG_STOPLEN_2 (0x01 << 6)
- #define UART_CFG_MODE32K (0x01 << 7)
- #define UART_CFG_LINMODE (0x01 << 8)
- #define UART_CFG_CTSEN (0x01 << 9)
- #define UART_CFG_SYNCEN (0x01 << 11)
- #define UART_CFG_CLKPOL (0x01 << 12)
- #define UART_CFG_SYNCMST (0x01 << 14)
- #define UART_CFG_LOOP (0x01 << 15)
- #define UART_CFG_IOMODE (0x01 << 16)
- #define UART_CFG_OETA (0x01 << 18)
- #define UART_CFG_AUTOADDR (0x01 << 19)
- #define UART_CFG_OESEL (0x01 << 20)
- #define UART_CFG_OEPOL (0x01 << 21)
- #define UART_CFG_RXPOL (0x01 << 22)
- #define UART_CFG_TXPOL (0x01 << 23)
- #define UART_CTRL_TXBRKEN (0x01 << 1)

  *UART CTRL register definitions.*
- #define UART_CTRL_ADDRDET (0x01 << 2)
- #define UART_CTRL_TXDIS (0x01 << 6)
- #define UART_CTRL_CC (0x01 << 8)
- #define UART_CTRL_CLRCCONRX (0x01 << 9)
- #define UART_CTRL_AUTOBAUD (0x01 << 16)
- #define UART_STAT_RXIDLE (0x01 << 1)

  *UART STAT register definitions.*
- #define UART_STAT_TXIDLE (0x01 << 3)
- #define UART_STAT_CTS (0x01 << 4)
- #define UART_STAT_DELTACTS (0x01 << 5)
- #define UART_STAT_TXDISINT (0x01 << 6)
- #define UART_STAT_RXBRK (0x01 << 10)
- #define UART_STAT_DELTARXBRK (0x01 << 11)
- #define UART_STAT_START (0x01 << 12)
- #define UART_STAT_FRM_ERRINT (0x01 << 13)
- #define UART_STAT_PAR_ERRINT (0x01 << 14)
- #define UART_STAT_RXNOISEINT (0x01 << 15)
- #define UART_STAT_ABERR (0x01 << 16)
- #define UART_INT_TXIDLE (0x01 << 3)

  *UART INTENSET/INTENCLR/INTSTAT register definitions.*
- #define UART_INT_DELTACTS (0x01 << 5)
- #define UART_INT_TXDIS (0x01 << 6)
- #define UART_INT_DELTARXBRK (0x01 << 11)

- #define UART_INT_START (0x01 << 12)
- #define UART_INT_FRAMERR (0x01 << 13)
- #define UART_INT_PARITYERR (0x01 << 14)
- #define UART_INT_RXNOISE (0x01 << 15)
- #define UART_INT_ABERR (0x01 << 16)
- #define UART_FIFOCFG_BITMASK (0x7F033)

  *UART FIFO Configuration register bits.*
- #define UART_FIFOCFG_ENABLETX (1 << 0)
- #define UART_FIFOCFG_ENABLERX (1 << 1)
- #define UART_FIFOCFG_DMATX (1 << 12)
- #define UART_FIFOCFG_DMARX (1 << 13)
- #define UART_FIFOCFG_WAKETX (1 << 14)
- #define UART_FIFOCFG_WAKERX (1 << 15)
- #define UART_FIFOCFG_EMPTYTX (1 << 16)
- #define UART_FIFOCFG_EMPTYRX (1 << 17)
- #define UART_FIFO_DEPTH (16) /∗∗ UART-FIFO How many entries are in the FIFO ∗/

  *UART FIFO Status register defines.*
- #define UART_FIFOSTAT_BITMASK (0x1F1FFB) /∗∗ UART-FIFO STAT Register BitMask ∗/
- #define UART_FIFOSTAT_TXERR (1 << 0)
- #define UART_FIFOSTAT_RXERR (1 << 1)
- #define UART_FIFOSTAT_PERIPH (1 << 3)
- #define UART_FIFOSTAT_TXEMPTY (1 << 4)
- #define UART_FIFOSTAT_TXNOTFULL (1 << 5)
- #define UART_FIFOSTAT_RXNOTEMPTY (1 << 6)
- #define UART_FIFOSTAT_RXFULL (1 << 7)
- #define UART_FIFOSTAT_TXLVL(lvl) (((lvl) >> 8) & 0x1F)
- #define UART_FIFOSTAT_RXLVL(lvl) (((lvl) >> 16) & 0x1F)
- #define UART_FIFOTRIG_BITMASK (0x000f0f03) /∗∗ UART FIFO trigger settings Register BitMask ∗/

  *UART FIFO trigger settings register defines.*
- #define UART_FIFOTRIG_TXLVLENA (1 << 0)
- #define UART_FIFOTRIG_RXLVLENA (1 << 1)
- #define UART_FIFOTRIG_TXLVL(lvl) ((lvl & 0x0f) << 8)
- #define UART_FIFOTRIG_RXLVL(lvl) ((lvl & 0x0f) << 16)
- #define UART_FIFOINT_BITMASK (0x001F) /∗∗ FIFO interrupt Bit mask ∗/

  *UART FIFO Interrupt enable/disable/status [INTSET/INTCLR/INTSTAT and FIFOINTSET/FIFOINTCLR/FIFOINTS↩ TAT registers].*
- #define UART_FIFOINT_TXERR (1 << 0) /∗∗ TX error interrupt [BIT-0 of FIFOINTENSET/FIFOINTENCL↩ R/FIFOINTSTAT register] ∗/
- #define UART_FIFOINT_RXERR (1 << 1) /∗∗ RX error interrupt [BIT-1 of FIFOINTENSET/FIFOINTENC↩ LR/FIFOINTSTAT register] ∗/
- #define UART_FIFOINT_TXLVL (1 << 2) /∗∗ TX FIFO ready interrupt [BIT-2 of FIFOINTENSET/FIFOINT↩ ENCLR/FIFOINTSTAT register] ∗/
- #define UART_FIFOINT_RXLVL (1 << 3) /∗∗ RX Data ready interrupt [BIT-3 of FIFOINTENSET/FIFOINT↩ ENCLR/FIFOINTSTAT register] ∗/
- #define UART_FIFOINT_PERINT (1 << 4) /∗∗ UART peripheral interrupt [BIT-4 of FIFOINTSTAT register] ∗/

## Functions

- __STATIC_INLINE void Chip_UART_Enable (LPC_USART_T ∗pUART)

  *Enable the UART.*
- __STATIC_INLINE void Chip_UART_Disable (LPC_USART_T ∗pUART)

  *Disable the UART.*
- __STATIC_INLINE void Chip_UART_TXEnable (LPC_USART_T ∗pUART)

*Enable transmission on UART TxD pin.*

- __STATIC_INLINE void Chip_UART_TXDisable (LPC_USART_T ∗pUART)

    *Disable transmission on UART TxD pin.*
- __STATIC_INLINE uint32_t Chip_UART_AutoBaud (LPC_USART_T ∗pUART)

    *Set auto baud.*
- __STATIC_INLINE void Chip_UART_SendByte (LPC_USART_T ∗pUART, uint8_t data)

    *Transmit a single data byte through the UART peripheral.*
- __STATIC_INLINE uint32_t Chip_UART_ReadByte (LPC_USART_T ∗pUART)

    *Read a single byte data from the UART peripheral.*
- __STATIC_INLINE void Chip_UART_IntEnable (LPC_USART_T ∗pUART, uint32_t intMask)

    *Enable UART interrupts.*
- __STATIC_INLINE void Chip_UART_IntDisable (LPC_USART_T ∗pUART, uint32_t intMask)

    *Disable UART interrupts.*
- __STATIC_INLINE uint32_t Chip_UART_GetIntsEnabled (LPC_USART_T ∗pUART)

    *Returns UART interrupts that are enabled.*
- __STATIC_INLINE uint32_t Chip_UART_GetIntStatus (LPC_USART_T ∗pUART)

    *Get UART interrupt status.*
- __STATIC_INLINE void Chip_UART_ConfigData (LPC_USART_T ∗pUART, uint32_t config)

    *Configure data width, parity and stop bits.*
- __STATIC_INLINE uint32_t Chip_UART_GetStatus (LPC_USART_T ∗pUART)

    *Get the UART status register.*
- __STATIC_INLINE void Chip_UART_ClearStatus (LPC_USART_T ∗pUART, uint32_t stsMask)

    *Clear the UART status register.*
- __STATIC_INLINE uint32_t Chip_UART_GetFIFOStatus (LPC_USART_T ∗pUART)

    *Get the current status of UART controller FIFO.*
- __STATIC_INLINE void Chip_UART_ClearFIFOStatus (LPC_USART_T ∗pUART, uint32_t mask)

    *Clear the FIFO status register.*
- __STATIC_INLINE void Chip_UART_SetFIFOTrigLevel (LPC_USART_T ∗pUART, uint8_t tx_lvl, uint8_t rx←_lvl)

    *Setup the trigger level for UART FIFO.*
- __STATIC_INLINE void Chip_UART_EnableFIFOInts (LPC_USART_T ∗pUART, uint32_t intMask)

    *Enable a UART FIFO interrupt.*
- __STATIC_INLINE void Chip_UART_DisableFIFOInts (LPC_USART_T ∗pUART, uint32_t intMask)

    *Disable a UART FIFO interrupt.*
- __STATIC_INLINE uint32_t Chip_UART_GetFIFOEnabledInts (LPC_USART_T ∗pUART)

    *Return enabled UART FIFO interrupts.*
- __STATIC_INLINE uint32_t Chip_UART_GetFIFOPendingInts (LPC_USART_T ∗pUART)

    *Return pending UART FIFO interrupts.*
- __STATIC_INLINE void Chip_UART_SetFIFOCfg (LPC_USART_T ∗pUART, uint32_t cfg)

    *Set FIFO Configuration register.*
- __STATIC_INLINE void Chip_UART_ClearFIFOCfg (LPC_USART_T ∗pUART, uint32_t cfg)

    *Clear FIFO Configuration register.*
- __STATIC_INLINE void Chip_UART_FlushFIFOs (LPC_USART_T ∗pUART)

    *Flush FIFOs.*
- int Chip_UART_Init (LPC_USART_T ∗pUART)

    *Initialize the UART peripheral.*
- void Chip_UART_DeInit (LPC_USART_T ∗pUART)

    *Deinitialize the UART peripheral.*
- void Chip_UART_ConfigDMA (LPC_USART_T ∗pUART)

    *Configure UART for DMA.*
- int Chip_UART_Send (LPC_USART_T ∗pUART, const void ∗data, int numBytes)

*Transmit a byte array through the UART peripheral (non-blocking)*

- int Chip_UART_Read (LPC_USART_T ∗pUART, void ∗data, int numBytes)

    *Read data through the UART peripheral (non-blocking)*

- uint32_t Chip_UART_SetBaud (LPC_USART_T ∗pUART, uint32_t baudrate)

    *Set baud rate for UART.*

- int Chip_UART_SendBlocking (LPC_USART_T ∗pUART, const void ∗data, int numBytes)

    *Transmit a byte array through the UART peripheral (blocking)*

- int Chip_UART_ReadBlocking (LPC_USART_T ∗pUART, void ∗data, int numBytes)

    *Read data through the UART peripheral (blocking)*

- void Chip_UART_RXIntHandlerRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB)

    *UART receive-only interrupt handler for ring buffers.*

- void Chip_UART_TXIntHandlerRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB)

    *UART transmit-only interrupt handler for ring buffers.*

- uint32_t Chip_UART_SendRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB, const void ∗data, int count)

    *Populate a transmit ring buffer and start UART transmit.*

- int Chip_UART_ReadRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB, void ∗data, int bytes)

    *Copy data from a receive ring buffer.*

- void Chip_UART_IRQHandlerRB (LPC_USART_T ∗pUART, UART_STATISTICS_T ∗statistics, RINGBUFF_T ∗pRXRB, RINGBUFF_T ∗pTXRB)

    *UART receive/transmit interrupt handler for ring buffers.*

- void Chip_UART_IRQHandlerDMA (LPC_USART_T ∗pUART, UART_STATISTICS_T ∗statistics)

    *UART receive/transmit interrupt handler for DMA.*

## 8.52 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/usbd.h File Reference

### 8.52.1 Detailed Description

Common definitions and declarations for the USB stack.

Common definitions and declarations for the USB stack. `#include "lpc_types.h"`

### Data Structures

- struct WB_T
- union WORD_BYTE
- struct BM_T
- union REQUEST_TYPE
- struct USB_SETUP_PACKET
- struct USB_DEVICE_DESCRIPTOR
- struct USB_DEVICE_QUALIFIER_DESCRIPTOR
- struct USB_CONFIGURATION_DESCRIPTOR
- struct USB_IAD_DESCRIPTOR
- struct USB_INTERFACE_DESCRIPTOR
- struct USB_ENDPOINT_DESCRIPTOR
- struct USB_STRING_DESCRIPTOR
- struct USB_COMMON_DESCRIPTOR
- struct USB_OTHER_SPEED_CONFIGURATION

## Macros

- #define USB_CONFIG_POWER_MA(mA)  ((mA)/2)
- #define USB_ENDPOINT_0_HS_MAXP 64
- #define USB_ENDPOINT_0_LS_MAXP 8
- #define USB_ENDPOINT_BULK_HS_MAXP 512
- #define WBVAL(x) ((x) & 0xFF),(((x) >> 8) & 0xFF)
- #define B3VAL(x) ((x) & 0xFF),(((x) >> 8) & 0xFF),(((x) >> 16) & 0xFF)
- #define USB_DEVICE_DESC_SIZE (sizeof(USB_DEVICE_DESCRIPTOR))
- #define USB_CONFIGURATION_DESC_SIZE (sizeof(USB_CONFIGURATION_DESCRIPTOR))
- #define USB_INTERFACE_DESC_SIZE (sizeof(USB_INTERFACE_DESCRIPTOR))
- #define USB_INTERFACE_ASSOC_DESC_SIZE (sizeof(USB_IAD_DESCRIPTOR))
- #define USB_ENDPOINT_DESC_SIZE (sizeof(USB_ENDPOINT_DESCRIPTOR))
- #define USB_DEVICE_QUALI_SIZE (sizeof(USB_DEVICE_QUALIFIER_DESCRIPTOR))
- #define USB_OTHER_SPEED_CONF_SIZE (sizeof(USB_OTHER_SPEED_CONFIGURATION))

- #define REQUEST_HOST_TO_DEVICE 0
- #define REQUEST_DEVICE_TO_HOST 1

- #define REQUEST_STANDARD 0
- #define REQUEST_CLASS 1
- #define REQUEST_VENDOR 2
- #define REQUEST_RESERVED 3

- #define REQUEST_TO_DEVICE 0
- #define REQUEST_TO_INTERFACE 1
- #define REQUEST_TO_ENDPOINT 2
- #define REQUEST_TO_OTHER 3

- #define USB_REQUEST_GET_STATUS 0
- #define USB_REQUEST_CLEAR_FEATURE 1
- #define USB_REQUEST_SET_FEATURE 3
- #define USB_REQUEST_SET_ADDRESS 5
- #define USB_REQUEST_GET_DESCRIPTOR 6
- #define USB_REQUEST_SET_DESCRIPTOR 7
- #define USB_REQUEST_GET_CONFIGURATION 8
- #define USB_REQUEST_SET_CONFIGURATION 9
- #define USB_REQUEST_GET_INTERFACE 10
- #define USB_REQUEST_SET_INTERFACE 11
- #define USB_REQUEST_SYNC_FRAME 12

- #define USB_GETSTATUS_SELF_POWERED 0x01
- #define USB_GETSTATUS_REMOTE_WAKEUP 0x02
- #define USB_GETSTATUS_ENDPOINT_STALL 0x01

- #define USB_FEATURE_ENDPOINT_STALL 0
- #define USB_FEATURE_REMOTE_WAKEUP 1
- #define USB_FEATURE_TEST_MODE 2

- #define USB_DEVICE_DESCRIPTOR_TYPE 1
- #define USB_CONFIGURATION_DESCRIPTOR_TYPE 2
- #define USB_STRING_DESCRIPTOR_TYPE 3
- #define USB_INTERFACE_DESCRIPTOR_TYPE 4
- #define USB_ENDPOINT_DESCRIPTOR_TYPE 5

- #define USB_DEVICE_QUALIFIER_DESCRIPTOR_TYPE 6
- #define USB_OTHER_SPEED_CONFIG_DESCRIPTOR_TYPE 7
- #define USB_INTERFACE_POWER_DESCRIPTOR_TYPE 8
- #define USB_OTG_DESCRIPTOR_TYPE 9
- #define USB_DEBUG_DESCRIPTOR_TYPE 10
- #define USB_INTERFACE_ASSOCIATION_DESCRIPTOR_TYPE 11

- #define USB_DEVICE_CLASS_RESERVED 0x00
- #define USB_DEVICE_CLASS_AUDIO 0x01
- #define USB_DEVICE_CLASS_COMMUNICATIONS 0x02
- #define USB_DEVICE_CLASS_HUMAN_INTERFACE 0x03
- #define USB_DEVICE_CLASS_MONITOR 0x04
- #define USB_DEVICE_CLASS_PHYSICAL_INTERFACE 0x05
- #define USB_DEVICE_CLASS_POWER 0x06
- #define USB_DEVICE_CLASS_PRINTER 0x07
- #define USB_DEVICE_CLASS_STORAGE 0x08
- #define USB_DEVICE_CLASS_HUB 0x09
- #define USB_DEVICE_CLASS_MISCELLANEOUS 0xEF
- #define USB_DEVICE_CLASS_APP 0xFE
- #define USB_DEVICE_CLASS_VENDOR_SPECIFIC 0xFF

- #define USB_CONFIG_POWERED_MASK 0x40
- #define USB_CONFIG_BUS_POWERED 0x80
- #define USB_CONFIG_SELF_POWERED 0xC0
- #define USB_CONFIG_REMOTE_WAKEUP 0x20

- #define USB_ENDPOINT_DIRECTION_MASK 0x80
- #define USB_ENDPOINT_OUT(addr) ((addr) | 0x00)
- #define USB_ENDPOINT_IN(addr) ((addr) | 0x80)

- #define USB_ENDPOINT_TYPE_MASK 0x03
- #define USB_ENDPOINT_TYPE_CONTROL 0x00
- #define USB_ENDPOINT_TYPE_ISOCHRONOUS 0x01
- #define USB_ENDPOINT_TYPE_BULK 0x02
- #define USB_ENDPOINT_TYPE_INTERRUPT 0x03
- #define USB_ENDPOINT_SYNC_MASK 0x0C
- #define USB_ENDPOINT_SYNC_NO_SYNCHRONIZATION 0x00
- #define USB_ENDPOINT_SYNC_ASYNCHRONOUS 0x04
- #define USB_ENDPOINT_SYNC_ADAPTIVE 0x08
- #define USB_ENDPOINT_SYNC_SYNCHRONOUS 0x0C
- #define USB_ENDPOINT_USAGE_MASK 0x30
- #define USB_ENDPOINT_USAGE_DATA 0x00
- #define USB_ENDPOINT_USAGE_FEEDBACK 0x10
- #define USB_ENDPOINT_USAGE_IMPLICIT_FEEDBACK 0x20
- #define USB_ENDPOINT_USAGE_RESERVED 0x30

## Typedefs

- typedef void ∗ USBD_HANDLE_T

## 8.53 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/utick_5411x.h File Reference

### Data Structures

- struct LPC_UTICK_T

  *Micro Tick register block structure.*

### Macros

- #define UTICK_CTRL_REPEAT ((uint32_t) 1UL << 31)

  *UTick register definitions.*
- #define UTICK_CTRL_DELAY_MASK ((uint32_t) 0x7FFFFFFF)
- #define UTICK_STATUS_INTR ((uint32_t) 1 << 0)
- #define UTICK_STATUS_ACTIVE ((uint32_t) 1 << 1)
- #define UTICK_STATUS_MASK ((uint32_t) 0x03)

### Functions

- __STATIC_INLINE void Chip_UTICK_Init (LPC_UTICK_T ∗pUTICK)

  *Initialize the UTICK peripheral.*
- __STATIC_INLINE void Chip_UTICK_DeInit (LPC_UTICK_T ∗pUTICK)

  *De-initialize the UTICK peripheral.*
- __STATIC_INLINE void Chip_UTICK_SetTick (LPC_UTICK_T ∗pUTICK, uint32_t tick_value, bool repeat)

  *Setup UTICK.*
- __STATIC_INLINE void Chip_UTICK_SetDelayMs (LPC_UTICK_T ∗pUTICK, uint32_t delayMs, bool repeat)

  *Setup UTICK for the passed delay (in mS)*
- __STATIC_INLINE uint32_t Chip_UTICK_GetTick (LPC_UTICK_T ∗pUTICK)

  *Read UTICK Value.*
- __STATIC_INLINE void Chip_UTICK_Halt (LPC_UTICK_T ∗pUTICK)

  *Halt UTICK timer.*
- __STATIC_INLINE uint32_t Chip_UTICK_GetStatus (LPC_UTICK_T ∗pUTICK)

  *Returns the status of UTICK.*
- __STATIC_INLINE void Chip_UTICK_ClearInterrupt (LPC_UTICK_T ∗pUTICK)

  *Clears UTICK Interrupt flag.*

## 8.54 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/inc/wwdt_5411x.h File Reference

### Data Structures

- struct LPC_WWDT_T

  *Windowed Watchdog register block structure.*

### Macros

- #define WWDT_WDMOD_BITMASK ((uint32_t) 0x3F)

  *Watchdog Mode register definitions.*
- #define WWDT_WDMOD_WDEN ((uint32_t) (1 << 0))
- #define WWDT_WDMOD_WDRESET ((uint32_t) (1 << 1))
- #define WWDT_WDMOD_WDTOF ((uint32_t) (1 << 2))
- #define WWDT_WDMOD_WDINT ((uint32_t) (1 << 3))
- #define WWDT_WDMOD_WDPROTECT ((uint32_t) (1 << 4))
- #define WWDT_WDMOD_LOCK ((uint32_t) (1 << 5))

### Functions

- __STATIC_INLINE void Chip_WWDT_Init (LPC_WWDT_T ∗pWWDT)

  *Initialize the Watchdog timer.*
- __STATIC_INLINE void Chip_WWDT_DeInit (LPC_WWDT_T ∗pWWDT)

  *Shutdown the Watchdog timer.*
- __STATIC_INLINE void Chip_WWDT_SetTimeOut (LPC_WWDT_T ∗pWWDT, uint32_t timeout)

  *Set WDT timeout constant value used for feed.*
- __STATIC_INLINE void Chip_WWDT_Feed (LPC_WWDT_T ∗pWWDT)

  *Feed watchdog timer.*
- __STATIC_INLINE void Chip_WWDT_SetWarning (LPC_WWDT_T ∗pWWDT, uint32_t timeout)

  *Set WWDT warning interrupt.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetWarning (LPC_WWDT_T ∗pWWDT)

  *Get WWDT warning interrupt.*
- __STATIC_INLINE void Chip_WWDT_SetWindow (LPC_WWDT_T ∗pWWDT, uint32_t timeout)

  *Set WWDT window time.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetWindow (LPC_WWDT_T ∗pWWDT)

  *Get WWDT window time.*
- __STATIC_INLINE void Chip_WWDT_SetOption (LPC_WWDT_T ∗pWWDT, uint32_t options)

  *Enable watchdog timer options.*
- __STATIC_INLINE void Chip_WWDT_UnsetOption (LPC_WWDT_T ∗pWWDT, uint32_t options)

  *Disable/clear watchdog timer options.*
- __STATIC_INLINE void Chip_WWDT_Start (LPC_WWDT_T ∗pWWDT)

  *Enable WWDT activity.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetStatus (LPC_WWDT_T ∗pWWDT)

  *Read WWDT status flag.*
- __STATIC_INLINE void Chip_WWDT_ClearStatusFlag (LPC_WWDT_T ∗pWWDT, uint32_t status)

  *Clear WWDT interrupt status flags.*
- __STATIC_INLINE uint32_t Chip_WWDT_GetCurrentCount (LPC_WWDT_T ∗pWWDT)

  *Get the current value of WDT.*

## 8.55 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/adc_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- void Chip_ADC_Init (LPC_ADC_T ∗pADC, uint32_t flags)

  *Initialize the ADC peripheral.*
- void Chip_ADC_DeInit (LPC_ADC_T ∗pADC)

  *Shutdown ADC.*
- uint32_t Chip_ADC_Calibration (LPC_ADC_T ∗pADC)

  *Perform ADC calibration.*
- void Chip_ADC_SetClockRate (LPC_ADC_T ∗pADC, uint32_t rate)

  *Set ADC clock rate.*

## 8.56 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/chip_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- void SystemCoreClockUpdate (void)

  *Update system core and ASYNC syscon clock rate, should be called if the system has a clock rate change.*
- void Chip_USB_Init (void)

  *Initialize the USB bus.*

### Variables

- uint32_t SystemCoreClock

  *Current system clock rate, mainly used for peripherals in SYSCON.*

## 8.57 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/clock_5411x.c File Reference

```
#include "chip.h"
```

### Macros

- #define WDT_FREQ_LOOKUP

### Functions

- static uint32_t Chip_Clock_GetAsyncSyscon_ClockRate_NoDiv (void)
- uint32_t Chip_Clock_GetMain_A_ClockRate (void)

  *Return main A clock rate.*
- uint32_t Chip_Clock_GetMain_B_ClockRate (void)

  *Return main B clock rate.*
- void Chip_Clock_EnablePeriphClock (CHIP_SYSCON_CLOCK_T clk)

  *Enable a system or peripheral clock.*
- void Chip_Clock_DisablePeriphClock (CHIP_SYSCON_CLOCK_T clk)

  *Disable a system or peripheral clock.*
- uint32_t Chip_Clock_GetSysTickClockRate (void)

  *Returns the system tick rate as used with the system tick divider.*
- uint32_t Chip_Clock_GetADCClockRate (void)

  *Return ADC clock rate.*
- uint32_t Chip_Clock_GetAsyncSyscon_ClockRate (void)

  *Return asynchronous APB clock rate.*
- CHIP_SYSCON_MAINCLKSRC_T Chip_Clock_GetMainClockSource (void)

  *Get main system clock source.*
- uint32_t Chip_Clock_GetMainClockRate (void)

  *Return main clock rate.*
- uint32_t Chip_Clock_GetSystemClockRate (void)

  *Return system clock rate.*

- uint32_t Chip_Clock_GetFRGClockRate (void)

  *Get Fraction Rate Generator (FRG) clock rate.*
- uint32_t Chip_Clock_GetFRGInClockRate (void)

  *Get the input clock frequency of FRG.*
- uint32_t Chip_Clock_SetFRGClockRate (uint32_t rate)

  *Set FRG rate to given rate.*
- uint32_t Chip_Clock_GetMCLKClockRate (void)
- uint32_t Chip_Clock_GetFLEXCOMMClockRate (uint32_t id)

  *Return FlexCOMM clock rate.*
- uint32_t Chip_Clock_GetWDTOSCRate (void)

  *Return estimated watchdog oscillator rate.*

## Variables

- WEAK uint32_t mclk_in_rate = 0

## 8.57.1 Macro Definition Documentation

### 8.57.1.1 #define WDT_FREQ_LOOKUP

**Value:**

```
"\x0\x0F\x1E\x28\x3C\x46\x50\x5A\x64\x6E"\
 "\x78\x82\x8C\x91\x96\xA0\xA5\xAA\xB4\xB9"\
 "\xBE\xC8\xCD\xD2\xDC\xE1\xE6\xEB\xF0\xF5"
```

Definition at line 39 of file clock_5411x.c.

## 8.57.2 Function Documentation

### 8.57.2.1 static uint32_t Chip_Clock_GetAsyncSyscon_ClockRate_NoDiv ( void ) [static]

Definition at line 54 of file clock_5411x.c.

### 8.57.2.2 uint32_t Chip_Clock_GetMCLKClockRate ( void )

< HF-FRO 48MHz or 96MHz

< Main pll

< MCLK INPUT Clock pin set by IOCON

< Disable clock source to MCLK

Definition at line 350 of file clock_5411x.c.

## 8.57.3 Variable Documentation

### 8.57.3.1 WEAK uint32_t mclk_in_rate = 0

Definition at line 47 of file clock_5411x.c.

## 8.58 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/dma_5411x.c File Reference

```
#include "chip.h"
```

## 8.59 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/dma_service_5411x.c File Reference

```
#include "chip.h"
```

### Macros

- #define DMA_XFERCFG_WIDTH 8
- #define DMA_XFERCFG_SRCINC 12
- #define DMA_XFERCFG_DSTINC 14

### Functions

- void Chip_DMASERVICE_Init (DMA_CHDESC_T ∗base)

  *Initialize DMA service.*
- void Chip_DMASERVICE_ErrorHandler (void)
- void Chip_DMASERVICE_Isr (void)

  *DMA service interrupt handler.*
- void Chip_DMASERVICE_RegisterCb (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, DMA_CALLBACK_T pCallback)

  *Register callback function.*
- void Chip_DMASERVICE_SingleBuffer (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, uint32_t pMem, uint32_t length)

  *Use Single buffer mechanism.*
- void Chip_DMASERVICE_DoubleBuffer (const DMA_PERIPHERAL_CONTEXT_T ∗pContext, uint32_t pMem, uint32_t length, DMA_DUAL_DESCRIPTOR_T ∗pD)

  *Use double buffer mechanism.*

### Variables

- DMA_CALLBACK_T dma_pCallback_array [MAX_DMA_CHANNEL]
- DMA_CALLBACK_T dma_service_error_cb

### 8.59.1 Macro Definition Documentation

#### 8.59.1.1 #define DMA_XFERCFG_DSTINC 14

Definition at line 97 of file dma_service_5411x.c.

#### 8.59.1.2 #define DMA_XFERCFG_SRCINC 12

Definition at line 96 of file dma_service_5411x.c.

#### 8.59.1.3 #define DMA_XFERCFG_WIDTH 8

Definition at line 95 of file dma_service_5411x.c.

### 8.59.2 Function Documentation

#### 8.59.2.1 void Chip_DMASERVICE_ErrorHandler ( void )

Definition at line 50 of file dma_service_5411x.c.

### 8.59.3 Variable Documentation

#### 8.59.3.1 DMA_CALLBACK_T dma_pCallback_array[MAX_DMA_CHANNEL]

Definition at line 34 of file dma_service_5411x.c.

#### 8.59.3.2 DMA_CALLBACK_T dma_service_error_cb

Definition at line 35 of file dma_service_5411x.c.

## 8.60 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/dmic_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- void Chip_DMIC_Init (const CHIP_SYSCON_CLOCK_T clock, const CHIP_SYSCON_PERIPH_RESET_T reset)

  *Initialize DMIC interface.*
- void Chip_DMIC_CfgIO (LPC_DMIC_T ∗pDMIC, DMIC_IO_T cfg)

  *Configure DMIC io.*
- void Chip_DMIC_SetOpMode (LPC_DMIC_T ∗pDMIC, OP_MODE_T mode)

  *Set DMIC operating mode.*
- void Chip_DMIC_CfgChannel (LPC_DMIC_T ∗pDMIC, uint32_t channel, DMIC_CHANNEL_CONFIG_T ∗channel_cfg)

  *Configure DMIC channel.*
- void Chip_DMIC_CfgChannelDc (LPC_DMIC_T ∗pDMIC, uint32_t channel, DC_REMOVAL_T dc_cut_level, uint32_t post_dc_gain_reduce, bool saturate16bit)

  *Configure DMIC channel DC removal setting.*
- void Chip_DMIC_Use2fs (LPC_DMIC_T ∗pDMIC, bool use2fs)

  *Configure Clock scaling.*
- void Chip_DMIC_EnableChannnel (LPC_DMIC_T ∗pDMIC, uint32_t channelmask)

  *Configure Clock scaling.*
- void Chip_DMIC_FifoChannel (LPC_DMIC_T ∗pDMIC, uint32_t channel, uint32_t trig_level, uint32_t enable, uint32_t resetn)

  *Configure fifo settings for DMIC channel.*

### Variables

- DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch0_dma_interleaved_context
- DMA_PERIPHERAL_CONTEXT_T dmic_ch1_dma_interleaved_context

## 8.61 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/flexcomm_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- int Chip_FLEXCOMM_GetIndex (LPC_FLEXCOMM_T ∗pFCOMM)

  *Get index of the FLEXCOMM corresponding to the given base address.*
- int Chip_FLEXCOMM_Init (LPC_FLEXCOMM_T ∗pFCOMM, FLEXCOMM_PERIPH_T periph)

  *Initialize FlexCOMM and associate it with a given peripheral.*
- void Chip_FLEXCOMM_DeInit (LPC_FLEXCOMM_T ∗pFCOMM)

  *Uninitialize the FlexCOMM.*
- int Chip_FLEXCOMM_SetPeriph (LPC_FLEXCOMM_T ∗pFCOMM, FLEXCOMM_PERIPH_T periph, int lock)

  *Set FLEXCOMM to a peripheral function.*

## 8.62 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/fpu_init.c File Reference

## 8.63 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/i2cm_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- void Chip_I2CM_SetBusSpeed (LPC_I2C_T ∗pI2C, uint32_t busSpeed)

  *Set up bus speed for LPC_I2C controller.*
- uint32_t Chip_I2CM_XferHandler (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transfer state change handler.*
- void Chip_I2CM_Xfer (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transmit and Receive data in master mode.*
- uint32_t Chip_I2CM_XferBlocking (LPC_I2C_T ∗pI2C, I2CM_XFER_T ∗xfer)

  *Transmit and Receive data in master mode.*

## 8.64 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/i2cs_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- uint32_t Chip_I2CS_XferHandler (LPC_I2C_T ∗pI2C, const I2CS_XFER_T ∗xfers)

  *Slave transfer state change handler.*

## 8.65 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/i2s_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- int Chip_I2S_Init (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt)

  *Initialize I2S driver.*
- Status Chip_I2S_Config (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt)

  *Configure I2S Port.*
- void Chip_I2S_FIFO_Config (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt)

  *Configure I2S FIFO.*
- static void fifo_ctrl_tx (LPC_I2S_T ∗pI2S, I2S_FIFO_CMD_T cmd)
- static void fifo_ctrl_rx (LPC_I2S_T ∗pI2S, I2S_FIFO_CMD_T cmd)
- void Chip_I2S_FIFO_Control (LPC_I2S_T ∗pI2S, I2S_AUDIO_FORMAT_T ∗fmt, I2S_FIFO_CMD_T cmd)

  *Execute I2S FIFO control commands.*
- void Chip_I2S_ErrorHandler (LPC_I2S_T ∗pI2S, I2S_STATISTICS_T ∗stat)

  *I2S error handler.*

### 8.65.1 Function Documentation

#### 8.65.1.1 Status Chip_I2S_Config ( LPC_I2S_T ∗ *pI2S,* I2S_AUDIO_FORMAT_T ∗ *fmt* )

Configure I2S Port.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *fmt* | : Audio Format |

**Returns**

SUCCESS or ERROR

Definition at line 67 of file i2s_5411x.c.

#### 8.65.1.2 void Chip_I2S_ErrorHandler ( LPC_I2S_T ∗ *pI2S,* I2S_STATISTICS_T ∗ *stat* )

I2S error handler.

**Parameters**

| | |
|---|---|
| *I2S* | : The base of the I2S peripheral on the chip |

| | |
|---|---|
| *stat* | : Statistics structure |

**Returns**

Nothing

Definition at line 210 of file i2s_5411x.c.

### 8.65.1.3 void Chip_I2S_FIFO_Config ( LPC_I2S_T ∗ *pI2S,* I2S_AUDIO_FORMAT_T ∗ *fmt* )

Configure I2S FIFO.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *fmpt* | : Audio format information |

**Returns**

Nothing

Definition at line 120 of file i2s_5411x.c.

### 8.65.1.4 void Chip_I2S_FIFO_Control ( LPC_I2S_T ∗ *pI2S,* I2S_AUDIO_FORMAT_T ∗ *fmt,* I2S_FIFO_CMD_T *cmd* )

Execute I2S FIFO control commands.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base I2S peripheral on the chip |
| *fmt* | : Audio format information |
| *cmd* | : FIFO command |

**Returns**

Nothing

Definition at line 191 of file i2s_5411x.c.

### 8.65.1.5 int Chip_I2S_Init ( LPC_I2S_T ∗ *pI2S,* I2S_AUDIO_FORMAT_T ∗ *fmt* )

Initialize I2S driver.

**Parameters**

| | |
|---|---|
| *pI2S* | : The base of I2S peripheral on the chip |

**Returns**

0 - on success; ERR_FLEXCOMM_FUNCNOTSUPPORTED or ERR_FLEXCOMM_NOTFREE on failure

Definition at line 49 of file i2s_5411x.c.

### 8.65.1.6 static void fifo_ctrl_rx ( LPC_I2S_T ∗ *pI2S,* I2S_FIFO_CMD_T *cmd* ) [static]

Definition at line 167 of file i2s_5411x.c.

#### 8.65.1.7 static void fifo_ctrl_tx ( LPC_I2S_T ∗ *pI2S,* I2S_FIFO_CMD_T *cmd* ) [static]

Definition at line 140 of file i2s_5411x.c.

## 8.66 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/iap.c File Reference

```
#include "chip.h"
```

### Functions

- uint8_t Chip_IAP_PreSectorForReadWrite (uint32_t strSector, uint32_t endSector)

  *Prepare sector for write operation.*
- uint8_t Chip_IAP_CopyRamToFlash (uint32_t dstAdd, uint32_t ∗srcAdd, uint32_t byteswrt)

  *Copy RAM to flash.*
- uint8_t Chip_IAP_EraseSector (uint32_t strSector, uint32_t endSector)

  *Erase sector.*
- uint8_t Chip_IAP_BlankCheckSector (uint32_t strSector, uint32_t endSector)

  *Blank check a sector or multiples sector of on-chip flash memory.*
- uint32_t Chip_IAP_ReadPID ()

  *Read part identification number.*
- uint8_t Chip_IAP_ReadBootCode ()

  *Read boot code version number.*
- uint8_t Chip_IAP_Compare (uint32_t dstAdd, uint32_t srcAdd, uint32_t bytescmp)

  *Compare the memory contents at two locations.*
- uint8_t Chip_IAP_ReinvokeISP ()

  *IAP reinvoke ISP to invoke the bootloader in ISP mode.*
- uint32_t Chip_IAP_ReadUID ()

  *Read the unique ID.*
- uint8_t Chip_IAP_ErasePage (uint32_t strPage, uint32_t endPage)

  *Erase a page or multiple papers of on-chip flash memory.*

## 8.67 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/pll_5411x.c File Reference

```
#include "chip.h"
```

### Macros

- #define NVALMAX (0x100)
- #define PVALMAX (0x20)
- #define MVALMAX (0x8000)
- #define SYS_PLL_SELR(d) (((d) & 0xf) << 0)
- #define SYS_PLL_SELI(d) (((d) & 0x3f) << 4)
- #define SYS_PLL_SELP(d) (((d) & 0x1f) << 10)
- #define SYS_PLL_BYPASS (1 << 15)
- #define SYS_PLL_BYPASSCCODIV2 (1 << 16)
- #define SYS_PLL_UPLIMOFF (1 << 17)
- #define SYS_PLL_BANDSEL (1 << 18)

- #define SYS_PLL_DIRECTI (1 << 19)
- #define SYS_PLL_DIRECTO (1 << 20)
- #define PLL_SSCG0_MDEC_VAL_P (0)
- #define PLL_SSCG0_MDEC_VAL_M (0x1FFFFUL << PLL_SSCG0_MDEC_VAL_P)
- #define PLL_NDEC_VAL_P (0)
- #define PLL_NDEC_VAL_M (0x3FFUL << PLL_NDEC_VAL_P)
- #define PLL_PDEC_VAL_P (0)
- #define PLL_PDEC_VAL_M (0x3FFUL << PLL_PDEC_VAL_P)
- #define PLL_MIN_CCO_FREQ_MHZ (75000000)
- #define PLL_MAX_CCO_FREQ_MHZ (150000000)
- #define PLL_LOWER_IN_LIMIT (4000)
- #define PLL_MIN_IN_SSMODE (2000000)
- #define PLL_MAX_IN_SSMODE (4000000)
- #define PLL_SSCG_MF_FREQ_VALUE 4
- #define PLL_SSCG_MC_COMP_VALUE 2
- #define PLL_SSCG_MR_DEPTH_VALUE 4
- #define PLL_SSCG_DITHER_VALUE 0
- #define SYSCON_SYSPLLCTRL_SELR_P 0
- #define SYSCON_SYSPLLCTRL_SELR_M (0xFUL << SYSCON_SYSPLLCTRL_SELR_P)
- #define SYSCON_SYSPLLCTRL_SELI_P 4
- #define SYSCON_SYSPLLCTRL_SELI_M (0x3FUL << SYSCON_SYSPLLCTRL_SELI_P)
- #define SYSCON_SYSPLLCTRL_SELP_P 10
- #define SYSCON_SYSPLLCTRL_SELP_M (0x1FUL << SYSCON_SYSPLLCTRL_SELP_P)
- #define SYSCON_SYSPLLCTRL_BYPASS_P 15
- #define SYSCON_SYSPLLCTRL_BYPASS (1UL << SYSCON_SYSPLLCTRL_BYPASS_P)
- #define SYSCON_SYSPLLCTRL_BYPASS_FBDIV2_P 16
- #define SYSCON_SYSPLLCTRL_BYPASS_FBDIV2 (1UL << SYSCON_SYSPLLCTRL_BYPASS_FBDIV2_P)
- #define SYSCON_SYSPLLCTRL_UPLIMOFF_P 17
- #define SYSCON_SYSPLLCTRL_UPLIMOFF (1UL << SYSCON_SYSPLLCTRL_UPLIMOFF_P)
- #define SYSCON_SYSPLLCTRL_BANDSEL_SSCGREG_N_P 18
- #define SYSCON_SYSPLLCTRL_BANDSEL_SSCGREG_N (1UL << SYSCON_SYSPLLCTRL_BANDSEL_SSCGREG_N_P)
- #define SYSCON_SYSPLLCTRL_DIRECTI_P 19
- #define SYSCON_SYSPLLCTRL_DIRECTI (1UL << SYSCON_SYSPLLCTRL_DIRECTI_P)
- #define SYSCON_SYSPLLCTRL_DIRECTO_P 20
- #define SYSCON_SYSPLLCTRL_DIRECTO (1UL << SYSCON_SYSPLLCTRL_DIRECTO_P)
- #define SYSCON_SYSPLLSTAT_LOCK_P 0
- #define SYSCON_SYSPLLSTAT_LOCK (1UL << SYSCON_SYSPLLSTAT_LOCK_P)
- #define PLL_CTRL_BYPASS_P 15
- #define PLL_CTRL_BYPASS_FBDIV2_P 16
- #define PLL_CTRL_UPLIMOFF_P 17
- #define PLL_CTRL_BANDSEL_SSCGREG_N_P 18
- #define PLL_CTRL_DIRECTI_P 19
- #define PLL_CTRL_DIRECTO_P 20
- #define PLL_CTRL_BYPASS (1 << PLL_CTRL_BYPASS_P)
- #define PLL_CTRL_DIRECTI (1 << PLL_CTRL_DIRECTI_P)
- #define PLL_CTRL_DIRECTO (1 << PLL_CTRL_DIRECTO_P)
- #define PLL_CTRL_UPLIMOFF (1 << PLL_CTRL_UPLIMOFF_P)
- #define PLL_CTRL_BANDSEL_SSCGREG_N (1 << PLL_CTRL_BANDSEL_SSCGREG_N_P)
- #define PLL_CTRL_BYPASS_FBDIV2 (1 << PLL_CTRL_BYPASS_FBDIV2_P)
- #define PLL_SSCG0_MREQ_P 17
- #define PLL_SSCG0_SEL_EXT_SSCG_N_P 18
- #define PLL_SSCG0_SEL_EXT_SSCG_N (1 << PLL_SSCG0_SEL_EXT_SSCG_N_P)
- #define PLL_SSCG0_MREQ (1 << PLL_SSCG0_MREQ_P)

- #define PLL_SSCG1_MD_REQ_P 19
- #define PLL_SSCG1_MOD_PD_SSCGCLK_N_P 28
- #define PLL_SSCG1_DITHER_P 29
- #define PLL_SSCG1_MOD_PD_SSCGCLK_N (1 << PLL_SSCG1_MOD_PD_SSCGCLK_N_P)
- #define PLL_SSCG1_DITHER (1 << PLL_SSCG1_DITHER_P)
- #define PLL_SSCG1_MD_REQ (1 << PLL_SSCG1_MD_REQ_P)
- #define PLL_NDEC_VAL_SET(value) (((unsigned long) (value) << PLL_NDEC_VAL_P) & PLL_NDEC_VAL_M)
- #define PLL_NDEC_NREQ_P 10
- #define PLL_NDEC_NREQ (1 << PLL_NDEC_NREQ_P)
- #define PLL_PDEC_VAL_SET(value) (((unsigned long) (value) << PLL_PDEC_VAL_P) & PLL_PDEC_VAL_M)
- #define PLL_PDEC_PREQ_P 7
- #define PLL_PDEC_PREQ (1 << PLL_PDEC_PREQ_P)
- #define PLL_SSCG0_MDEC_VAL_SET(value) (((unsigned long) (value) << PLL_SSCG0_MDEC_VAL_P) & PLL_SSCG0_MDEC_VAL_M)
- #define PLL_SSCG0_MREQ_P 17
- #define PLL_SSCG0_MREQ (1 << PLL_SSCG0_MREQ_P)
- #define PLL_SSCG0_SEL_EXT_SSCG_N_P 18
- #define PLL_SSCG0_SEL_EXT_SSCG_N (1 << PLL_SSCG0_SEL_EXT_SSCG_N_P)
- #define PLL_SSCG1_MD_FRACT_P 0
- #define PLL_SSCG1_MD_INT_P 11
- #define PLL_SSCG1_MF_P 20
- #define PLL_SSCG1_MC_P 26
- #define PLL_SSCG1_MR_P 23
- #define PLL_SSCG1_MD_FRACT_M (0x7FFUL << PLL_SSCG1_MD_FRACT_P)
- #define PLL_SSCG1_MD_INT_M (0xFFUL << PLL_SSCG1_MD_INT_P)
- #define PLL_SSCG1_MF_M (0x7UL << PLL_SSCG1_MF_P)
- #define PLL_SSCG1_MC_M (0x3UL << PLL_SSCG1_MC_P)
- #define PLL_SSCG1_MR_M (0x7UL << PLL_SSCG1_MR_P)
- #define PLL_SSCG1_MD_FRACT_SET(value)
- #define PLL_SSCG1_MD_INT_SET(value)
- #define PLL0_SSCG_MF_FREQ_VALUE 4
- #define PLL0_SSCG_MC_COMP_VALUE 2
- #define PLL0_SSCG_MR_DEPTH_VALUE 4
- #define PLL0_SSCG_DITHER_VALUE 0
- #define PLL_MAX_N_DIV 0x100

## Functions

- static uint32_t pllEncodeN (uint32_t N)
- static uint32_t pllDecodeN (uint32_t NDEC)
- static uint32_t pllEncodeP (uint32_t P)
- static uint32_t pllDecodeP (uint32_t PDEC)
- static uint32_t pllEncodeM (uint32_t M)
- static uint32_t pllDecodeM (uint32_t MDEC)
- static void pllFindSel (uint32_t M, bool bypassFBDIV2, uint32_t ∗pSelP, uint32_t ∗pSelI, uint32_t ∗pSelR)
- uint32_t findPllPreDiv (uint32_t ctrlReg, uint32_t nDecReg)
- uint32_t findPllPostDiv (uint32_t ctrlReg, uint32_t pDecReg)
- uint32_t findPllMMult (uint32_t ctrlReg, uint32_t mDecReg)
- static uint32_t FindGreatestCommonDivisor (uint32_t m, uint32_t n)
- static PLL_ERROR_T Chip_Clock_GetPllConfig (uint32_t finHz, uint32_t foutHz, PLL_SETUP_T ∗pSetup, bool useFeedbackDiv2, bool useSS)
- static void Chip_Clock_GetSystemPLLOutFromSetupUpdate (PLL_SETUP_T ∗pSetup)

- uint32_t Chip_Clock_GetSystemPLLInClockRate (void)

  *Return System PLL input clock rate.*
- uint32_t Chip_Clock_GetSystemPLLOutFromSetup (PLL_SETUP_T ∗pSetup)

  *Return System PLL output clock rate from setup structure.*
- uint32_t Chip_Clock_GetStoredPLLClockRate (void)

  *Get the rate of pll from the stored value.*
- void Chip_Clock_SetStoredPLLClockRate (uint32_t rate)

  *Store the current PLL rate.*
- uint32_t Chip_Clock_GetSystemPLLOutClockRate (bool recompute)

  *Return System PLL output clock rate.*
- void Chip_Clock_SetBypassPLL (bool bypass)

  *Enables and disables PLL bypass mode.*
- PLL_ERROR_T Chip_Clock_SetupPLLData (PLL_CONFIG_T ∗pControl, PLL_SETUP_T ∗pSetup)

  *Set PLL output based on the passed PLL setup data.*
- PLL_ERROR_T Chip_Clock_SetupSystemPLLPrec (PLL_SETUP_T ∗pSetup)

  *Set PLL output from PLL setup structure (precise frequency)*
- PLL_ERROR_T Chip_Clock_SetPLLFreq (const PLL_SETUP_T ∗pSetup)

  *Set PLL output from PLL setup structure (precise frequency)*
- void Chip_Clock_SetupSystemPLL (uint32_t multiply_by, uint32_t input_freq)

  *Set PLL output based on the multiplier and input frequency.*

## Variables

- static uint32_t curPllRate

### 8.67.1 Macro Definition Documentation

#### 8.67.1.1 #define MVALMAX (0x8000)

Definition at line 40 of file pll_5411x.c.

#### 8.67.1.2 #define NVALMAX (0x100)

Definition at line 38 of file pll_5411x.c.

#### 8.67.1.3 #define PLL0_SSCG_DITHER_VALUE 0

Definition at line 169 of file pll_5411x.c.

#### 8.67.1.4 #define PLL0_SSCG_MC_COMP_VALUE 2

Definition at line 167 of file pll_5411x.c.

#### 8.67.1.5 #define PLL0_SSCG_MF_FREQ_VALUE 4

Definition at line 166 of file pll_5411x.c.

#### 8.67.1.6 #define PLL0_SSCG_MR_DEPTH_VALUE 4

Definition at line 168 of file pll_5411x.c.

#### 8.67.1.7 #define PLL_CTRL_BANDSEL_SSCGREG_N (1 << PLL_CTRL_BANDSEL_SSCGREG_N_P)

Definition at line 107 of file pll_5411x.c.

#### 8.67.1.8 #define PLL_CTRL_BANDSEL_SSCGREG_N_P 18

Definition at line 99 of file pll_5411x.c.

#### 8.67.1.9 #define PLL_CTRL_BYPASS (1 << PLL_CTRL_BYPASS_P)

Definition at line 103 of file pll_5411x.c.

#### 8.67.1.10 #define PLL_CTRL_BYPASS_FBDIV2 (1 << PLL_CTRL_BYPASS_FBDIV2_P)

Definition at line 108 of file pll_5411x.c.

#### 8.67.1.11 #define PLL_CTRL_BYPASS_FBDIV2_P 16

Definition at line 97 of file pll_5411x.c.

#### 8.67.1.12 #define PLL_CTRL_BYPASS_P 15

Definition at line 96 of file pll_5411x.c.

#### 8.67.1.13 #define PLL_CTRL_DIRECTI (1 << PLL_CTRL_DIRECTI_P)

Definition at line 104 of file pll_5411x.c.

#### 8.67.1.14 #define PLL_CTRL_DIRECTI_P 19

Definition at line 100 of file pll_5411x.c.

#### 8.67.1.15 #define PLL_CTRL_DIRECTO (1 << PLL_CTRL_DIRECTO_P)

Definition at line 105 of file pll_5411x.c.

#### 8.67.1.16 #define PLL_CTRL_DIRECTO_P 20

Definition at line 101 of file pll_5411x.c.

#### 8.67.1.17 #define PLL_CTRL_UPLIMOFF (1 << PLL_CTRL_UPLIMOFF_P)

Definition at line 106 of file pll_5411x.c.

#### 8.67.1.18 #define PLL_CTRL_UPLIMOFF_P 17

Definition at line 98 of file pll_5411x.c.

#### 8.67.1.19 #define PLL_LOWER_IN_LIMIT (4000)

Minimum PLL input rate

Definition at line 63 of file pll_5411x.c.

#### 8.67.1.20 #define PLL_MAX_CCO_FREQ_MHZ (150000000)

Definition at line 62 of file pll_5411x.c.

#### 8.67.1.21 #define PLL_MAX_IN_SSMODE (4000000)

Definition at line 65 of file pll_5411x.c.

#### 8.67.1.22 #define PLL_MAX_N_DIV 0x100

Definition at line 171 of file pll_5411x.c.

#### 8.67.1.23 #define PLL_MIN_CCO_FREQ_MHZ (75000000)

Definition at line 61 of file pll_5411x.c.

#### 8.67.1.24 #define PLL_MIN_IN_SSMODE (2000000)

Definition at line 64 of file pll_5411x.c.

#### 8.67.1.25 #define PLL_NDEC_NREQ (1 << PLL_NDEC_NREQ_P)

Definition at line 128 of file pll_5411x.c.

#### 8.67.1.26 #define PLL_NDEC_NREQ_P 10

Definition at line 127 of file pll_5411x.c.

#### 8.67.1.27 #define PLL_NDEC_VAL_M (0x3FFUL << PLL_NDEC_VAL_P)

Definition at line 57 of file pll_5411x.c.

#### 8.67.1.28 #define PLL_NDEC_VAL_P (0)

Definition at line 56 of file pll_5411x.c.

#### 8.67.1.29 #define PLL_NDEC_VAL_SET( *value* ) (((unsigned long) (value) << PLL_NDEC_VAL_P) & PLL_NDEC_VAL_M)

Definition at line 126 of file pll_5411x.c.

#### 8.67.1.30 #define PLL_PDEC_PREQ (1 << PLL_PDEC_PREQ_P)

Definition at line 133 of file pll_5411x.c.

#### 8.67.1.31 #define PLL_PDEC_PREQ_P 7

Definition at line 132 of file pll_5411x.c.

#### 8.67.1.32 #define PLL_PDEC_VAL_M (0x3FFUL << PLL_PDEC_VAL_P)

Definition at line 59 of file pll_5411x.c.

#### 8.67.1.33 #define PLL_PDEC_VAL_P (0)

Definition at line 58 of file pll_5411x.c.

#### 8.67.1.34 #define PLL_PDEC_VAL_SET( *value* ) (((unsigned long) (value) << PLL_PDEC_VAL_P) & PLL_PDEC_VAL_M)

Definition at line 131 of file pll_5411x.c.

#### 8.67.1.35 #define PLL_SSCG0_MDEC_VAL_M (0x1FFFFUL << PLL_SSCG0_MDEC_VAL_P)

Definition at line 55 of file pll_5411x.c.

#### 8.67.1.36 #define PLL_SSCG0_MDEC_VAL_P (0)

Definition at line 54 of file pll_5411x.c.

#### 8.67.1.37 #define PLL_SSCG0_MDEC_VAL_SET( *value* ) (((unsigned long) (value) << PLL_SSCG0_MDEC_VAL_P) & PLL_SSCG0_MDEC_VAL_M)

Definition at line 136 of file pll_5411x.c.

#### 8.67.1.38 #define PLL_SSCG0_MREQ (1 << PLL_SSCG0_MREQ_P)

Definition at line 138 of file pll_5411x.c.

#### 8.67.1.39 #define PLL_SSCG0_MREQ (1 << PLL_SSCG0_MREQ_P)

Definition at line 138 of file pll_5411x.c.

#### 8.67.1.40 #define PLL_SSCG0_MREQ_P 17

Definition at line 137 of file pll_5411x.c.

#### 8.67.1.41 #define PLL_SSCG0_MREQ_P 17

Definition at line 137 of file pll_5411x.c.

#### 8.67.1.42 #define PLL_SSCG0_SEL_EXT_SSCG_N (1 << PLL_SSCG0_SEL_EXT_SSCG_N_P)

Definition at line 140 of file pll_5411x.c.

#### 8.67.1.43 #define PLL_SSCG0_SEL_EXT_SSCG_N (1 << PLL_SSCG0_SEL_EXT_SSCG_N_P)

Definition at line 140 of file pll_5411x.c.

#### 8.67.1.44 #define PLL_SSCG0_SEL_EXT_SSCG_N_P 18

Definition at line 139 of file pll_5411x.c.

#### 8.67.1.45 #define PLL_SSCG0_SEL_EXT_SSCG_N_P 18

Definition at line 139 of file pll_5411x.c.

#### 8.67.1.46 #define PLL_SSCG1_DITHER (1 << PLL_SSCG1_DITHER_P)

Definition at line 122 of file pll_5411x.c.

#### 8.67.1.47 #define PLL_SSCG1_DITHER_P 29

Definition at line 120 of file pll_5411x.c.

#### 8.67.1.48 #define PLL_SSCG1_MC_M (0x3UL << PLL_SSCG1_MC_P)

Definition at line 152 of file pll_5411x.c.

#### 8.67.1.49 #define PLL_SSCG1_MC_P 26

Definition at line 146 of file pll_5411x.c.

#### 8.67.1.50 #define PLL_SSCG1_MD_FRACT_M (0x7FFUL << PLL_SSCG1_MD_FRACT_P)

Definition at line 149 of file pll_5411x.c.

#### 8.67.1.51 #define PLL_SSCG1_MD_FRACT_P 0

Definition at line 143 of file pll_5411x.c.

#### 8.67.1.52 #define PLL_SSCG1_MD_FRACT_SET( *value* )

**Value:**

```
(((unsigned long) (value) << \
                    PLL_SSCG1_MD_FRACT_P) &
      PLL_SSCG1_MD_FRACT_M)
```

Definition at line 155 of file pll_5411x.c.

#### 8.67.1.53 #define PLL_SSCG1_MD_INT_M (0xFFUL << PLL_SSCG1_MD_INT_P)

Definition at line 150 of file pll_5411x.c.

#### 8.67.1.54 #define PLL_SSCG1_MD_INT_P 11

Definition at line 144 of file pll_5411x.c.

#### 8.67.1.55 #define PLL_SSCG1_MD_INT_SET( *value* )

**Value:**

```
(((unsigned long) (value) << \
                  PLL_SSCG1_MD_INT_P)    &
      PLL_SSCG1_MD_INT_M)
```

Definition at line 157 of file pll_5411x.c.

#### 8.67.1.56 #define PLL_SSCG1_MD_REQ (1 << PLL_SSCG1_MD_REQ_P)

Definition at line 123 of file pll_5411x.c.

#### 8.67.1.57 #define PLL_SSCG1_MD_REQ_P 19

Definition at line 118 of file pll_5411x.c.

#### 8.67.1.58 #define PLL_SSCG1_MF_M (0x7UL << PLL_SSCG1_MF_P)

Definition at line 151 of file pll_5411x.c.

#### 8.67.1.59 #define PLL_SSCG1_MF_P 20

Definition at line 145 of file pll_5411x.c.

#### 8.67.1.60 #define PLL_SSCG1_MOD_PD_SSCGCLK_N (1 << PLL_SSCG1_MOD_PD_SSCGCLK_N_P)

Definition at line 121 of file pll_5411x.c.

#### 8.67.1.61 #define PLL_SSCG1_MOD_PD_SSCGCLK_N_P 28

Definition at line 119 of file pll_5411x.c.

#### 8.67.1.62 #define PLL_SSCG1_MR_M (0x7UL << PLL_SSCG1_MR_P)

Definition at line 153 of file pll_5411x.c.

#### 8.67.1.63 #define PLL_SSCG1_MR_P 23

Definition at line 147 of file pll_5411x.c.

#### 8.67.1.64 #define PLL_SSCG_DITHER_VALUE 0

Definition at line 71 of file pll_5411x.c.

#### 8.67.1.65 #define PLL_SSCG_MC_COMP_VALUE 2

Definition at line 69 of file pll_5411x.c.

#### 8.67.1.66 #define PLL_SSCG_MF_FREQ_VALUE 4

Definition at line 68 of file pll_5411x.c.

#### 8.67.1.67 #define PLL_SSCG_MR_DEPTH_VALUE 4

Definition at line 70 of file pll_5411x.c.

#### 8.67.1.68 #define PVALMAX (0x20)

Definition at line 39 of file pll_5411x.c.

#### 8.67.1.69 #define SYS_PLL_BANDSEL (1 << 18)

Enable MDEC control

Definition at line 49 of file pll_5411x.c.

#### 8.67.1.70 #define SYS_PLL_BYPASS (1 << 15)

Enable PLL bypass

Definition at line 46 of file pll_5411x.c.

#### 8.67.1.71 #define SYS_PLL_BYPASSCCODIV2 (1 << 16)

Enable bypass of extra divider by 2

Definition at line 47 of file pll_5411x.c.

#### 8.67.1.72 #define SYS_PLL_DIRECTI (1 << 19)

PLL0 direct input enable

Definition at line 50 of file pll_5411x.c.

#### 8.67.1.73 #define SYS_PLL_DIRECTO (1 << 20)

PLL0 direct output enable

Definition at line 51 of file pll_5411x.c.

#### 8.67.1.74 #define SYS_PLL_SELI( *d* ) (((d) & 0x3f) << 4)

Bandwidth select I value

Definition at line 44 of file pll_5411x.c.

#### 8.67.1.75 #define SYS_PLL_SELP( *d* ) (((d) & 0x1f) << 10)

Bandwidth select P value

Definition at line 45 of file pll_5411x.c.

#### 8.67.1.76 #define SYS_PLL_SELR( *d* ) (((d) & 0xf) << 0)

Bandwidth select R value

Definition at line 43 of file pll_5411x.c.

#### 8.67.1.77 #define SYS_PLL_UPLIMOFF (1 << 17)

Enable spread spectrum/fractional mode

Definition at line 48 of file pll_5411x.c.

#### 8.67.1.78 #define SYSCON_SYSPLLCTRL_BANDSEL_SSCGREG_N (1UL << SYSCON_SYSPLLCTRL_BANDSEL_SSCGREG↩_N_P)

Definition at line 87 of file pll_5411x.c.

#### 8.67.1.79 #define SYSCON_SYSPLLCTRL_BANDSEL_SSCGREG_N_P 18

Definition at line 86 of file pll_5411x.c.

#### 8.67.1.80 #define SYSCON_SYSPLLCTRL_BYPASS (1UL << SYSCON_SYSPLLCTRL_BYPASS_P)

Definition at line 81 of file pll_5411x.c.

#### 8.67.1.81 #define SYSCON_SYSPLLCTRL_BYPASS_FBDIV2 (1UL << SYSCON_SYSPLLCTRL_BYPASS_FBDIV2_P)

Definition at line 83 of file pll_5411x.c.

#### 8.67.1.82 #define SYSCON_SYSPLLCTRL_BYPASS_FBDIV2_P 16

Definition at line 82 of file pll_5411x.c.

#### 8.67.1.83 #define SYSCON_SYSPLLCTRL_BYPASS_P 15

Definition at line 80 of file pll_5411x.c.

#### 8.67.1.84 #define SYSCON_SYSPLLCTRL_DIRECTI (1UL << SYSCON_SYSPLLCTRL_DIRECTI_P)

Definition at line 89 of file pll_5411x.c.

#### 8.67.1.85 #define SYSCON_SYSPLLCTRL_DIRECTI_P 19

Definition at line 88 of file pll_5411x.c.

#### 8.67.1.86 #define SYSCON_SYSPLLCTRL_DIRECTO (1UL << SYSCON_SYSPLLCTRL_DIRECTO_P)

Definition at line 91 of file pll_5411x.c.

#### 8.67.1.87 #define SYSCON_SYSPLLCTRL_DIRECTO_P 20

Definition at line 90 of file pll_5411x.c.

#### 8.67.1.88 #define SYSCON_SYSPLLCTRL_SELI_M (0x3FUL << SYSCON_SYSPLLCTRL_SELI_P)

Definition at line 77 of file pll_5411x.c.

#### 8.67.1.89 #define SYSCON_SYSPLLCTRL_SELI_P 4

Definition at line 76 of file pll_5411x.c.

#### 8.67.1.90 #define SYSCON_SYSPLLCTRL_SELP_M (0x1FUL << SYSCON_SYSPLLCTRL_SELP_P)

Definition at line 79 of file pll_5411x.c.

#### 8.67.1.91 #define SYSCON_SYSPLLCTRL_SELP_P 10

Definition at line 78 of file pll_5411x.c.

#### 8.67.1.92 #define SYSCON_SYSPLLCTRL_SELR_M (0xFUL << SYSCON_SYSPLLCTRL_SELR_P)

Definition at line 75 of file pll_5411x.c.

#### 8.67.1.93 #define SYSCON_SYSPLLCTRL_SELR_P 0

Definition at line 74 of file pll_5411x.c.

#### 8.67.1.94 #define SYSCON_SYSPLLCTRL_UPLIMOFF (1UL << SYSCON_SYSPLLCTRL_UPLIMOFF_P)

Definition at line 85 of file pll_5411x.c.

#### 8.67.1.95 #define SYSCON_SYSPLLCTRL_UPLIMOFF_P 17

Definition at line 84 of file pll_5411x.c.

#### 8.67.1.96 #define SYSCON_SYSPLLSTAT_LOCK (1UL << SYSCON_SYSPLLSTAT_LOCK_P)

Definition at line 94 of file pll_5411x.c.

#### 8.67.1.97 #define SYSCON_SYSPLLSTAT_LOCK_P 0

Definition at line 93 of file pll_5411x.c.

### 8.67.2 Function Documentation

#### 8.67.2.1 static PLL_ERROR_T Chip_Clock_GetPllConfig ( uint32_t *finHz*, uint32_t *foutHz*, PLL_SETUP_T ∗ *pSetup*, bool *useFeedbackDiv2*, bool *useSS* ) [static]

Definition at line 488 of file pll_5411x.c.

#### 8.67.2.2 static void Chip_Clock_GetSystemPLLOutFromSetupUpdate ( PLL_SETUP_T ∗ *pSetup* ) [static]

Definition at line 643 of file pll_5411x.c.

#### 8.67.2.3 static uint32_t FindGreatestCommonDivisor ( uint32_t *m*, uint32_t *n* ) [static]

Definition at line 474 of file pll_5411x.c.

#### 8.67.2.4 uint32_t findPllMMult ( uint32_t *ctrlReg*, uint32_t *mDecReg* )

Definition at line 455 of file pll_5411x.c.

#### 8.67.2.5 uint32_t findPllPostDiv ( uint32_t *ctrlReg*, uint32_t *pDecReg* )

Definition at line 437 of file pll_5411x.c.

#### 8.67.2.6 uint32_t findPllPreDiv ( uint32_t *ctrlReg*, uint32_t *nDecReg* )

Definition at line 419 of file pll_5411x.c.

#### 8.67.2.7 static uint32_t pllDecodeM ( uint32_t *MDEC* ) [static]

Definition at line 346 of file pll_5411x.c.

#### 8.67.2.8 static uint32_t pllDecodeN ( uint32_t *NDEC* ) [static]

Definition at line 216 of file pll_5411x.c.

#### 8.67.2.9 static uint32_t pllDecodeP ( uint32_t *PDEC* ) [static]

Definition at line 281 of file pll_5411x.c.

#### 8.67.2.10 static uint32_t pllEncodeM ( uint32_t *M* ) [static]

Definition at line 316 of file pll_5411x.c.

#### 8.67.2.11 static uint32_t pllEncodeN ( uint32_t *N* ) [static]

Definition at line 186 of file pll_5411x.c.

#### 8.67.2.12 static uint32_t pllEncodeP ( uint32_t *P* ) [static]

Definition at line 251 of file pll_5411x.c.

#### 8.67.2.13 static void pllFindSel ( uint32_t *M,* bool *bypassFBDIV2,* uint32_t ∗ *pSelP,* uint32_t ∗ *pSelI,* uint32_t ∗ *pSelR* ) [static]

Definition at line 381 of file pll_5411x.c.

### 8.67.3 Variable Documentation

#### 8.67.3.1 uint32_t curPllRate [static]

Definition at line 175 of file pll_5411x.c.

## 8.68 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/ring_buffer.c File Reference

```
#include <string.h>
#include "ring_buffer.h"
```

### Macros

- #define RB_INDH(rb) ((rb)->head & ((rb)->count - 1))
- #define RB_INDT(rb) ((rb)->tail & ((rb)->count - 1))

### Functions

- int RingBuffer_Init (RINGBUFF_T ∗RingBuff, void ∗buffer, int itemSize, int count, void ∗(∗cpyFunc)(void ∗dst, const void ∗src, uint32_t len))

  *Initialize ring buffer.*
- int RingBuffer_Insert (RINGBUFF_T ∗RingBuff, const void ∗data)

  *Insert a single item into ring buffer.*
- int RingBuffer_InsertMult (RINGBUFF_T ∗RingBuff, const void ∗data, int num)

  *Insert an array of items into ring buffer.*
- int RingBuffer_Pop (RINGBUFF_T ∗RingBuff, void ∗data)

  *Pop an item from the ring buffer.*
- int RingBuffer_PopMult (RINGBUFF_T ∗RingBuff, void ∗data, int num)

  *Pop an array of items from the ring buffer.*

### 8.68.1 Macro Definition Documentation

#### 8.68.1.1 #define RB_INDH( *rb* ) ((rb)->head & ((rb)->count - 1))

Definition at line 39 of file ring_buffer.c.

#### 8.68.1.2 #define RB_INDT( *rb* ) ((rb)->tail & ((rb)->count - 1))

Definition at line 40 of file ring_buffer.c.

## 8.69 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/rtc_ut.c File Reference

```
#include "rtc_ut.h"
```

## Macros

- #define SECSPERMIN (60)
- #define MINSPERHOUR (60)
- #define SECSPERHOUR (SECSPERMIN ∗ MINSPERHOUR)
- #define HOURSPERDAY (24)
- #define SECSPERDAY (SECSPERMIN ∗ MINSPERHOUR ∗ HOURSPERDAY)
- #define DAYSPERWEEK (7)
- #define MONETHSPERYEAR (12)
- #define DAYSPERYEAR (365)
- #define DAYSPERLEAPYEAR (366)

## Functions

- static void GetDMLY (int dayOff, struct tm ∗pTime)
- void ConvertRtcTime (uint32_t rtcTick, struct tm ∗pTime)

  *Converts a RTC tick time to Universal time.*
- void ConvertTimeRtc (struct tm ∗pTime, uint32_t ∗rtcTick)

  *Converts a Universal time to RTC tick time.*

## Variables

- static uint8_t daysPerMonth [2][MONETHSPERYEAR]

## 8.69.1 Macro Definition Documentation

### 8.69.1.1 #define DAYSPERLEAPYEAR (366)

Definition at line 48 of file rtc_ut.c.

### 8.69.1.2 #define DAYSPERWEEK (7)

Definition at line 45 of file rtc_ut.c.

### 8.69.1.3 #define DAYSPERYEAR (365)

Definition at line 47 of file rtc_ut.c.

### 8.69.1.4 #define HOURSPERDAY (24)

Definition at line 43 of file rtc_ut.c.

### 8.69.1.5 #define MINSPERHOUR (60)

Definition at line 41 of file rtc_ut.c.

### 8.69.1.6 #define MONETHSPERYEAR (12)

Definition at line 46 of file rtc_ut.c.

#### 8.69.1.7 #define SECSPERDAY (SECSPERMIN ∗ MINSPERHOUR ∗ HOURSPERDAY)

Definition at line 44 of file rtc_ut.c.

#### 8.69.1.8 #define SECSPERHOUR (SECSPERMIN ∗ MINSPERHOUR)

Definition at line 42 of file rtc_ut.c.

#### 8.69.1.9 #define SECSPERMIN (60)

Definition at line 40 of file rtc_ut.c.

### 8.69.2 Function Documentation

#### 8.69.2.1 static void GetDMLY ( int *dayOff,* struct tm ∗ *pTime* ) [static]

Definition at line 66 of file rtc_ut.c.

### 8.69.3 Variable Documentation

#### 8.69.3.1 uint8_t daysPerMonth[2][MONETHSPERYEAR] [static]

**Initial value:**

```
= {
 {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
 {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
}
```

Definition at line 51 of file rtc_ut.c.

## 8.70 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/sct_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- void Chip_SCT_Init (LPC_SCT_T ∗pSCT)

  *Initializes the State Configurable Timer.*
- void Chip_SCT_DeInit (LPC_SCT_T ∗pSCT)

  *Deinitializes the State Configurable Timer.*
- void Chip_SCT_SetClrControl (LPC_SCT_T ∗pSCT, uint32_t value, FunctionalState ena)

  *Set or Clear the Control register.*
- void Chip_SCT_SetConflictResolution (LPC_SCT_T ∗pSCT, uint8_t outnum, uint8_t value)

  *Set the conflict resolution.*

## 8.71 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/sct_pwm_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- void Chip_SCTPWM_SetOutPin (LPC_SCT_T ∗pSCT, uint8_t index, uint8_t pin)

  *Setup the OUTPUT pin and associate it with an index.*
- void Chip_SCTPWM_SetRate (LPC_SCT_T ∗pSCT, uint32_t freq)

  *Sets the frequency of the generated PWM wave.*

## 8.72 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/spi_common_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- int Chip_SPI_Init (LPC_SPI_T ∗pSPI)

  *Initialize the SPI.*
- void Chip_SPI_ConfigureSPI (LPC_SPI_T ∗pSPI, SPI_CFGSETUP_T ∗pCFG)

  *Setup SPI configuration.*

## 8.73 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/spim_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- static uint16_t get_tx_data (SPIM_XFER_T ∗xfer)
- static void put_rx_data (SPIM_XFER_T ∗xfer, uint16_t data)
- static uint16_t spim_get_xfercfg (const SPIM_XFER_T ∗xfer)
- static void spim_do_txrx (LPC_SPI_T ∗pSPI, uint32_t stat, SPIM_XFER_T ∗xfer)
- uint32_t Chip_SPIM_SetClockRate (LPC_SPI_T ∗pSPI, uint32_t rate)

  *Set SPI master bit rate.*
- void Chip_SPIM_DelayConfig (LPC_SPI_T ∗pSPI, SPIM_DELAY_CONFIG_T ∗pConfig)

  *Config SPI Delay parameters.*
- void Chip_SPIM_XferHandler (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *SPI master transfer state change handler.*
- void Chip_SPIM_Xfer (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Start non-blocking SPI master transfer.*
- void Chip_SPIM_XferFIFO (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Start polled SPI master transfer.*
- void Chip_SPIM_XferBlocking (LPC_SPI_T ∗pSPI, SPIM_XFER_T ∗xfer)

  *Perform blocking SPI master transfer.*

### 8.73.1 Function Documentation

#### 8.73.1.1 static uint16_t get_tx_data ( SPIM_XFER_T ∗ *xfer* ) [static]

Definition at line 46 of file spim_5411x.c.

#### 8.73.1.2 static void put_rx_data ( SPIM_XFER_T ∗ *xfer,* uint16_t *data* ) [static]

Definition at line 66 of file spim_5411x.c.

#### 8.73.1.3 static void spim_do_txrx ( LPC_SPI_T ∗ *pSPI,* uint32_t *stat,* SPIM_XFER_T ∗ *xfer* ) [static]

Definition at line 101 of file spim_5411x.c.

#### 8.73.1.4 static uint16_t spim_get_xfercfg ( const SPIM_XFER_T ∗ *xfer* ) [static]

Definition at line 83 of file spim_5411x.c.

## 8.74 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/spis_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- static uint16_t get_tx_data (SPIS_XFER_T ∗xfer)
- static void put_rx_data (SPIS_XFER_T ∗xfer, uint16_t data)
- static void spis_do_txrx (LPC_SPI_T ∗pSPI, uint32_t stat, SPIS_XFER_T ∗xfer)
- void Chip_SPIS_Init (LPC_SPI_T ∗pSPI)

  *SPI slave initialization.*
- void Chip_SPIS_EnableInts (LPC_SPI_T ∗pSPI)

  *SPI slave interrupt enable.*
- void Chip_SPIS_DisableInts (LPC_SPI_T ∗pSPI)

  *SPI slave interrupt disable.*
- void Chip_SPIS_LoadFIFO (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *Load slave transmit FIFO.*
- void Chip_SPIS_ReadFIFO (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *SPI slave FIFO read.*
- void Chip_SPIS_XferHandler (LPC_SPI_T ∗pSPI, SPIS_XFER_T ∗xfer)

  *SPI slave transfer state change handler.*

### 8.74.1 Function Documentation

#### 8.74.1.1 static uint16_t get_tx_data ( SPIS_XFER_T ∗ *xfer* ) [static]

Definition at line 47 of file spis_5411x.c.

#### 8.74.1.2 static void put_rx_data ( SPIS_XFER_T ∗ *xfer,* uint16_t *data* ) [static]

Definition at line 68 of file spis_5411x.c.

#### 8.74.1.3 static void spis_do_txrx ( LPC_SPI_T ∗ *pSPI,* uint32_t *stat,* SPIS_XFER_T ∗ *xfer* ) [static]

Definition at line 85 of file spis_5411x.c.

## 8.75 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/stopwatch_5411x.c File Reference

```
#include "chip.h"
#include "stopwatch.h"
```

### Macros

- #define LPC_TIMER32_1 LPC_TIMER0

### Functions

- void StopWatch_Init (void)

  *Initialize stopwatch.*
- uint32_t StopWatch_Start (void)

  *Start a stopwatch.*
- uint32_t StopWatch_TicksPerSecond (void)

  *Returns number of ticks per second of the stopwatch timer.*
- uint32_t StopWatch_TicksToMs (uint32_t ticks)

  *Converts from stopwatch ticks to mS.*
- uint32_t StopWatch_TicksToUs (uint32_t ticks)

  *Converts from stopwatch ticks to uS.*
- uint32_t StopWatch_MsToTicks (uint32_t mS)

  *Converts from mS to stopwatch ticks.*
- uint32_t StopWatch_UsToTicks (uint32_t uS)

  *Converts from uS to stopwatch ticks.*

### Variables

- static uint32_t ticksPerSecond
- static uint32_t ticksPerMs
- static uint32_t ticksPerUs

### 8.75.1 Macro Definition Documentation

#### 8.75.1.1 #define LPC_TIMER32_1 LPC_TIMER0

Definition at line 45 of file stopwatch_5411x.c.

### 8.75.2 Variable Documentation

#### 8.75.2.1 uint32_t ticksPerMs [static]

Definition at line 41 of file stopwatch_5411x.c.

#### 8.75.2.2 uint32_t ticksPerSecond [static]

Definition at line 40 of file stopwatch_5411x.c.

#### 8.75.2.3 uint32_t ticksPerUs [static]

Definition at line 42 of file stopwatch_5411x.c.

## 8.76 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/syscon_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- void Chip_SYSCON_SetNMISource (uint32_t intsrc)

  *Set source for non-maskable interrupt (NMI)*
- void Chip_SYSCON_EnableNMISource (void)

  *Enable interrupt used for NMI source.*
- void Chip_SYSCON_DisableNMISource (void)

  *Disable interrupt used for NMI source.*
- void Chip_SYSCON_Enable_ASYNC_Syscon (bool enable)

  *Enable or disable asynchronous APB bridge and subsystem.*
- uint32_t Chip_SYSCON_GetCompFreqMeas (uint32_t refClockRate)

  *Returns the computed value for a frequency measurement cycle.*
- void Chip_SYSCON_PowerUp (uint32_t powerupmask)

  *Power up one or more blocks or peripherals.*

## 8.77 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/sysinit_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- static void setupFlashClocks (uint32_t freq)
- void Chip_SetupFROClocking (uint32_t iFreq)

  *Initialize the Core clock to given frequency (12, 48 or 96 MHz)*
- void Chip_SetupIrcClocking (uint32_t iFreq)

  *Clock and PLL initialization based on the internal oscillator.*
- void Chip_SetupExtInClocking (uint32_t iFreq)

  *Clock and PLL initialization based on the external clock input.*
- void Chip_SystemInit (void)

  *Set up and initialize hardware prior to call to main()*

### 8.77.1 Function Documentation

#### 8.77.1.1 static void setupFlashClocks ( uint32_t *freq* ) [static]

Definition at line 47 of file sysinit_5411x.c.

## 8.78 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/timer_5411x.c File Reference

```
#include "chip.h"
```

### Macros

- #define LAST_TIMER (4)

### Functions

- void Chip_TIMER_Reset (LPC_TIMER_T ∗pTMR)

  *Resets the timer terminal and prescale counts to 0.*
- void Chip_TIMER_ExtMatchControlSet (LPC_TIMER_T ∗pTMR, int8_t initial_state, TIMER_PIN_MATCH_STATE_T matchState, int8_t matchnum)

  *Sets external match control (MATn.matchnum) pin control. For the pin selected with matchnum, sets the function of the pin that occurs on a terminal count match for the match count.*

### 8.78.1 Macro Definition Documentation

#### 8.78.1.1 #define LAST_TIMER (4)

Definition at line 37 of file timer_5411x.c.

## 8.79 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/src/uart_5411x.c File Reference

```
#include "chip.h"
```

### Functions

- static uint32_t _UART_DivClk (uint32_t pclk, uint32_t m)
- static uint32_t _UART_GetHighDiv (uint32_t val, uint8_t strict)
- static int32_t _CalcErr (uint32_t n, uint32_t d, uint32_t ∗prev)
- static ErrorCode_t _UART_CalcDiv (UART_BAUD_T ∗ub)
- static void _UART_CalcMul (UART_BAUD_T ∗ub)
- int Chip_UART_Init (LPC_USART_T ∗pUART)

  *Initialize the UART peripheral.*
- void Chip_UART_DeInit (LPC_USART_T ∗pUART)

  *Deinitialize the UART peripheral.*
- void Chip_UART_ConfigDMA (LPC_USART_T ∗pUART)

  *Configure UART for DMA.*
- int Chip_UART_Send (LPC_USART_T ∗pUART, const void ∗data, int numBytes)

  *Transmit a byte array through the UART peripheral (non-blocking)*
- int Chip_UART_SendBlocking (LPC_USART_T ∗pUART, const void ∗data, int numBytes)

  *Transmit a byte array through the UART peripheral (blocking)*
- int Chip_UART_Read (LPC_USART_T ∗pUART, void ∗data, int numBytes)

  *Read data through the UART peripheral (non-blocking)*
- int Chip_UART_ReadBlocking (LPC_USART_T ∗pUART, void ∗data, int numBytes)

  *Read data through the UART peripheral (blocking)*

- int Chip_UART_CalculateBaud (UART_BAUD_T ∗baud)
- uint32_t Chip_UART_SetBaud (LPC_USART_T ∗pUART, uint32_t baudrate)

  *Set baud rate for UART.*
- void Chip_UART_RXIntHandlerRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB)

  *UART receive-only interrupt handler for ring buffers.*
- void Chip_UART_TXIntHandlerRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB)

  *UART transmit-only interrupt handler for ring buffers.*
- uint32_t Chip_UART_SendRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB, const void ∗data, int count)

  *Populate a transmit ring buffer and start UART transmit.*
- int Chip_UART_ReadRB (LPC_USART_T ∗pUART, RINGBUFF_T ∗pRB, void ∗data, int bytes)

  *Copy data from a receive ring buffer.*
- void Chip_UART_IRQHandlerRB (LPC_USART_T ∗pUART, UART_STATISTICS_T ∗stat, RINGBUFF_↩T ∗pRXRB, RINGBUFF_T ∗pTXRB)

  *UART receive/transmit interrupt handler for ring buffers.*
- void Chip_UART_IRQHandlerDMA (LPC_USART_T ∗pUART, UART_STATISTICS_T ∗stat)

  *UART receive/transmit interrupt handler for DMA.*

## Variables

- static const uint32_t fifo_config_int = (UART_FIFOCFG_ENABLETX | UART_FIFOCFG_ENABLERX)
- static const uint32_t fifo_config_dma = (UART_FIFOCFG_ENABLETX | UART_FIFOCFG_ENABLERX | U↩ART_FIFOCFG_DMATX | UART_FIFOCFG_DMARX)

## 8.79.1 Function Documentation

### 8.79.1.1 static int32_t _CalcErr ( uint32_t *n*, uint32_t *d*, uint32_t ∗ *prev* ) [static]

Definition at line 70 of file uart_5411x.c.

### 8.79.1.2 static ErrorCode_t _UART_CalcDiv ( UART_BAUD_T ∗ *ub* ) [static]

Definition at line 86 of file uart_5411x.c.

### 8.79.1.3 static void _UART_CalcMul ( UART_BAUD_T ∗ *ub* ) [static]

Definition at line 124 of file uart_5411x.c.

### 8.79.1.4 static uint32_t _UART_DivClk ( uint32_t *pclk*, uint32_t *m* ) [static]

Definition at line 48 of file uart_5411x.c.

### 8.79.1.5 static uint32_t _UART_GetHighDiv ( uint32_t *val*, uint8_t *strict* ) [static]

Definition at line 58 of file uart_5411x.c.

### 8.79.1.6 int Chip_UART_CalculateBaud ( UART_BAUD_T ∗ *baud* )

Definition at line 242 of file uart_5411x.c.

### 8.79.2 Variable Documentation

#### 8.79.2.1 const uint32_t fifo_config_dma = (UART_FIFOCFG_ENABLETX | UART_FIFOCFG_ENABLERX | UART_FIFOCFG_DMATX | UART_FIFOCFG_DMARX) [static]

Definition at line 38 of file uart_5411x.c.

#### 8.79.2.2 const uint32_t fifo_config_int = (UART_FIFOCFG_ENABLETX | UART_FIFOCFG_ENABLERX) [static]

Definition at line 37 of file uart_5411x.c.

## 8.80 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/startup/cr_startup_lpc5411x-m0.c File Reference

### Macros

- #define WEAK __attribute__ ((weak))
- #define ALIAS(f) __attribute__ ((weak, alias (#f)))

### Functions

- void ResetISR (void)
- WEAK void NMI_Handler (void)
- WEAK void HardFault_Handler (void)
- WEAK void SVC_Handler (void)
- WEAK void PendSV_Handler (void)
- WEAK void SysTick_Handler (void)
- WEAK void IntDefaultHandler (void)
- void WDT_BOD_IRQHandler (void DMA_IRQHandler() ALIAS(IntDefaultHandler) void)
- __attribute__ ((section(".after_vectors")))
- __attribute__ ((section(".after_vectors.reset")))

### Variables

- unsigned int __data_section_table
- unsigned int __data_section_table_end
- unsigned int __bss_section_table
- unsigned int __bss_section_table_end

### 8.80.1 Macro Definition Documentation

#### 8.80.1.1 #define ALIAS( *f* ) __attribute__ ((weak, alias (#f)))

Definition at line 48 of file cr_startup_lpc5411x-m0.c.

#### 8.80.1.2 #define WEAK __attribute__ ((weak))

Definition at line 47 of file cr_startup_lpc5411x-m0.c.

### 8.80.2 Function Documentation

#### 8.80.2.1 __attribute__ ( (section(".after_vectors")) )

Definition at line 234 of file cr_startup_lpc5411x-m0.c.

#### 8.80.2.2 __attribute__ ( (section(".after_vectors.reset")) )

Definition at line 328 of file cr_startup_lpc5411x-m0.c.

#### 8.80.2.3 WEAK void HardFault_Handler ( void )

#### 8.80.2.4 WEAK void IntDefaultHandler ( void )

#### 8.80.2.5 WEAK void NMI_Handler ( void )

#### 8.80.2.6 WEAK void PendSV_Handler ( void )

#### 8.80.2.7 void ResetISR ( void )

#### 8.80.2.8 WEAK void SVC_Handler ( void )

#### 8.80.2.9 WEAK void SysTick_Handler ( void )

#### 8.80.2.10 void WDT_BOD_IRQHandler ( void DMA_IRQHandler() ALIAS(IntDefaultHandler) *void* )

Definition at line 92 of file cr_startup_lpc5411x-m0.c.

### 8.80.3 Variable Documentation

#### 8.80.3.1 unsigned int __bss_section_table

#### 8.80.3.2 unsigned int __bss_section_table_end

#### 8.80.3.3 unsigned int __data_section_table

#### 8.80.3.4 unsigned int __data_section_table_end

## 8.81 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/startup/cr_startup_lpc5411x.c File Reference

### Macros

- #define WEAK __attribute__ ((weak))
- #define ALIAS(f) __attribute__ ((weak, alias (#f)))

### Functions

- void ResetISR (void)
- WEAK void NMI_Handler (void)
- WEAK void HardFault_Handler (void)
- WEAK void MemManage_Handler (void)

- WEAK void BusFault_Handler (void)
- WEAK void UsageFault_Handler (void)
- WEAK void SVC_Handler (void)
- WEAK void DebugMon_Handler (void)
- WEAK void PendSV_Handler (void)
- WEAK void SysTick_Handler (void)
- WEAK void IntDefaultHandler (void)
- void WDT_BOD_IRQHandler (void DMA_IRQHandler() ALIAS(IntDefaultHandler) void)
- __attribute__ ((section(".after_vectors")))
- __attribute__ ((section(".after_vectors.reset")))

## Variables

- unsigned int __SWVtrace_Enabled
- unsigned int __data_section_table
- unsigned int __data_section_table_end
- unsigned int __bss_section_table
- unsigned int __bss_section_table_end

## 8.81.1 Macro Definition Documentation

### 8.81.1.1 #define ALIAS( *f* ) __attribute__ ((weak, alias (#f)))

Definition at line 48 of file cr_startup_lpc5411x.c.

### 8.81.1.2 #define WEAK __attribute__ ((weak))

Definition at line 47 of file cr_startup_lpc5411x.c.

## 8.81.2 Function Documentation

### 8.81.2.1 __attribute__ ( (section(".after_vectors")) )

Definition at line 248 of file cr_startup_lpc5411x.c.

### 8.81.2.2 __attribute__ ( (section(".after_vectors.reset")) )

Definition at line 340 of file cr_startup_lpc5411x.c.

### 8.81.2.3 WEAK void BusFault_Handler ( void )

### 8.81.2.4 WEAK void DebugMon_Handler ( void )

### 8.81.2.5 WEAK void HardFault_Handler ( void )

### 8.81.2.6 WEAK void IntDefaultHandler ( void )

### 8.81.2.7 WEAK void MemManage_Handler ( void )

### 8.81.2.8 WEAK void NMI_Handler ( void )

#### 8.81.2.9 WEAK void PendSV_Handler ( void )

#### 8.81.2.10 void ResetISR ( void )

#### 8.81.2.11 WEAK void SVC_Handler ( void )

#### 8.81.2.12 WEAK void SysTick_Handler ( void )

#### 8.81.2.13 WEAK void UsageFault_Handler ( void )

#### 8.81.2.14 void WDT_BOD_IRQHandler ( void DMA_IRQHandler() **ALIAS(IntDefaultHandler)** *void* )

Definition at line 98 of file cr_startup_lpc5411x.c.

### 8.81.3 Variable Documentation

#### 8.81.3.1 unsigned int __bss_section_table

#### 8.81.3.2 unsigned int __bss_section_table_end

#### 8.81.3.3 unsigned int __data_section_table

#### 8.81.3.4 unsigned int __data_section_table_end

#### 8.81.3.5 unsigned int __SWVtrace_Enabled

Definition at line 64 of file cr_startup_lpc5411x.c.

## 8.82 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/startup/crp.c File Reference

## 8.83 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/startup/mtb.c File Reference

## 8.84 C:/Jenkins/LPC5411x/lpc5411x/chip_5411x/startup/sysinit.c File Reference

```
#include "board.h"
```

### Functions

- void SystemInit (void)

### 8.84.1 Function Documentation

#### 8.84.1.1 void SystemInit ( void )

Definition at line 59 of file sysinit.c.

## 8.85 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/common/community_support.dox File Reference

## 8.86 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/common/copyright.dox File Reference

## 8.87 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/common/installation_insts.dox File Reference

## 8.88 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/common/toolchains.dox File Reference

## 8.89 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/common/version_history.dox File Reference

## 8.90 boards_5411x.dox File Reference

## 8.91 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/dox_lpc5411x/chip_5411x/chip_↩5411x.dox File Reference

## 8.92 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/dox_lpc5411x/chip_5411x/irq_vector↩_names.dox File Reference

## 8.93 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/dox_lpc5411x/chip_5411x/lpcopen_↩layout.dox File Reference

## 8.94 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/dox_lpc5411x/chip_5411x/main_↩layout.dox File Reference

## 8.95 C:/Jenkins/LPC5411x/lpcdocs/docs/doxygen/dox_lpc5411x/chip_5411x/multicore.dox File Reference

# Index