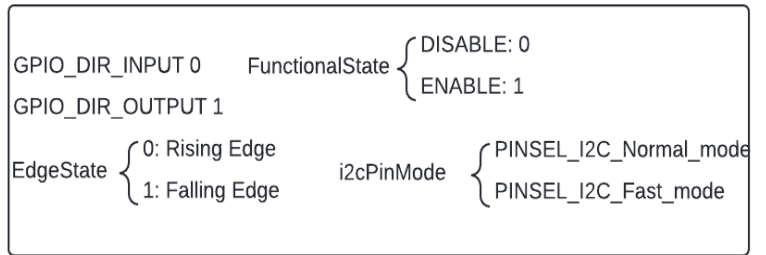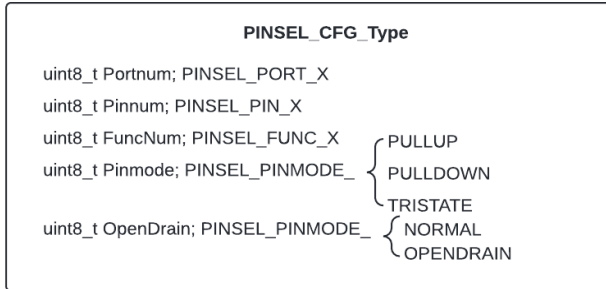- void **GPIO_SetDir**(uint8_t portNum, uint32_t bitValue, uint8_t dir);
- void **GPIO_SetValue**(uint8_t portNum, uint32_t bitValue);
- void **GPIO_ ClearValue** (uint8_t portNum, uint32_t bitValue);
- void **GPIO_ ReadValue** (uint8_t portNum);
- void **GPIO_IntCmd**(uint8_t portNum, uint32_t bitValue, uint8_t edgeState);
- void **GPIO_ClearInt**(uint8_t portNum, uint32_t bitValue);

"lpc17xx_pinsel.h"

- void **PINSEL_ConfigPin**(PINSEL_CFG_Type* PinCfg);
- void **PINSEL _ConfigTraceFunc**(FunctionalState NewState);
- void **PINSEL _ SetI2C0Pins**(uint8_t i2cPinMode, FunctionalState filterSlewRateEnable);

PINSEL_CFG_Type

uint8_t Portnum; PINSEL_PORT_X
uint8_t Pinnum; PINSEL_PIN_X
uint8_t FuncNum; PINSEL_FUNC_X
uint8_t Pinmode; PINSEL_PINMODE_ { PULLUP / PULLDOWN / TRISTATE
uint8_t OpenDrain; PINSEL_PINMODE_ { NORMAL / OPENDRAIN

GPIO_DIR_INPUT 0   FunctionalState { DISABLE: 0 / ENABLE: 1
GPIO_DIR_OUTPUT 1

EdgeState { 0: Rising Edge / 1: Falling Edge   i2cPinMode { PINSEL_I2C_Normal_mode / PINSEL_I2C_Fast_mode
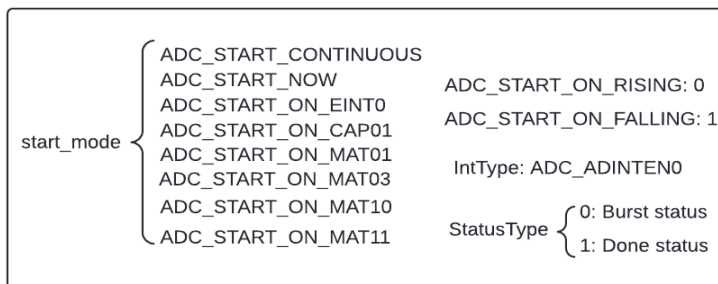
"lpc17xx_systick.h"

- void **SYSTICK_InternalInit**(uint32_t time);
- void **SYSTICK_ExternalInit**(uint32_t freq, uint32_t time);
- void **SYSTICK_Cmd**(FunctionalState NewState);
- void **SYSTICK_IntCmd**(FunctionalState NewState);
- uint32_t **SYSTICK_GetCurrent_Value**(void);
- void **SYSTICK_ClearCounterFlag**(void);

maximum time can be set:
   * 1/SystemCoreClock * (2^24) * 1000 (ms)

"lpc17xx_adc.h"

- void **ADC_Init**(LPC_ADC_TypeDef* ADCx, uint32_t rate); //   ADCx: LPC_ADC   rate:[Hz]
- void **ADC_DeInit**(LPC_ADC_TypeDef* ADCx);
- void **ADC_BurstCmd**(LPC_ADC_TypeDef* ADCx, FunctionalState NewState);
- void **ADC_PowerdownCmd**(LPC_ADC_TypeDef* ADCx, FunctionalState NewState);
- void **ADC_StartCmd**(LPC_ADC_TypeDef* ADCx, uint8_t start_mode);
- void **ADC_ChannelCmd**(LPC_ADC_TypeDef* ADCx, uint8_t Channel, FunctionalState NewState);
- void **ADC_EdgeStartConfig**(LPC_ADC_TypeDef* ADCx, uint8_t EdgeOption);
- void **ADC_IntConfig**(LPC_ADC_TypeDef* ADCx, ADC_TYPE_INT_OPT IntType, FunctionalState NewState);
- uint16_t **ADC_ChannelGetData**(LPC_ADC_TypeDef* ADCx, uint8_t channel);
- FlagStatus **ADC_ChannelGetStatus**(LPC_ADC_TypeDef* ADCx, uint8_t channel, uint32_t StatusType);
- uint32_t **ADC_GlobalGetData**(LPC_ADC_TypeDef* ADCx);
- FlagStatus **ADC_GlobalGetStatus**(LPC_ADC_TypeDef* ADCx, uint32_t StatusType);

start_mode {
ADC_START_CONTINUOUS
ADC_START_NOW
ADC_START_ON_EINT0
ADC_START_ON_CAP01
ADC_START_ON_MAT01
ADC_START_ON_MAT03
ADC_START_ON_MAT10
ADC_START_ON_MAT11
}

ADC_START_ON_RISING: 0
ADC_START_ON_FALLING: 1

IntType: ADC_ADINTEN0

StatusType { 0: Burst status / 1: Done status

"lpc17xx_timer.h"

- void **TIM_Init**(LPC_TIM_TypeDef* TIMx, TIM_MODE_OPT TimerCounterMode, void* TIM_ConfigStruct);
- void **TIM_DeInit**(LPC_TIM_TypeDef* TIMx);
- void **TIM_ClearIntPending**(LPC_TIM_TypeDef* TIMx, TIM_INT_TYPE IntFlag);
- void **TIM_ClearIntCapturePending**(LPC_TIM_TypeDef* TIMx, TIM_INT_TYPE IntFlag);
- FlagStatus **TIM_GetIntStatus**(LPC_TIM_TypeDef* TIMx, TIM_INT_TYPE IntFlag);
- FlagStatus **TIM_GetIntCaptureStatus**(LPC_TIM_TypeDef* TIMx, TIM_INT_TYPE IntFlag);
- void **TIM_ConfigStructInit**(TIM_MODE_OPT TimerCounterMode, void* TIM_ConfigStruct);
- void **TIM_ConfigMatch**(LPC_TIM_TypeDef* TIMx, TIM_MATCHCFG_Type* TIM_MatchConfigStruct);
- void **TIM_UpdateMatchValue**(LPC_TIM_TypeDef* TIMx, uint8_t MatchChannel, uint32_t MatchValue);
- void **TIM_SetMatchExt**(LPC_TIM_TypeDef* TIMx, TIM_EXTMATCH_OPT ext_match);
- void **TIM_ConfigCapture**(LPC_TIM_TypeDef* TIMx, TIM_CAPTURECFG_Type* TIM_CaptureConfigStruct);
- void **TIM_Cmd**(LPC_TIM_TypeDef* TIMx, FunctionalState NewState);
- uint32_t **TIM_GetCaptureValue**(LPC_TIM_TypeDef* TIMx, TIM_COUNTER_INPUT_OPT CaptureChannel);
- void **TIM_ResetCounter**(LPC_TIM_TypeDef* TIMx);

TIM_TIMERCFG_Type

uint8_t PrescaleOption; { TIM_PRESCALE_TICKVAL
                          TIM_PRESCALE_USVAL

uint8_t Reserved[3];

uint32_t RescaleValue;

TIM_MATCHCFG_Type

uint8_t MatchChannel;

uint8_t IntOnMatch;

uint8_t ResetOnMatch;

uint8_t ExtMatchOutputType; { TIM_EXMATCH_NOTHING
                              TIM_EXMATCH_LOW
                              TIM_EXMATCH_HIGH
                              TIM_EXMATCH_TOGGLE

uint8_t Reserved[3];

uint8_t MatchValue;

TIMx ⟶ LPC_TIMX "lpc17xx.h"

IntFlag { TIM_MR0_INT
          TIM_MR1_INT
          TIM_MR2_INT
          TIM_MR3_INT
          TIM_CR0_INT
          TIM_CR1_INT

TimerCounterMode { TIM_TIMER_MODE
                   TIM_COUNTER_RISING_MODE
                   TIM_COUNTER_FALLING_MODE
                   TIM_COUNTER_ANY_MODE