

## Anlogic Digital Conversor

Generated by Doxygen 1.9.1



<b>1 Class Index</b>	<b>1</b>
1.1 Class List	1
<b>2 File Index</b>	<b>3</b>
2.1 File List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 adc_chan Struct Reference	5
3.1.1 Detailed Description	6
3.2 adc_ctrl_t Struct Reference	6
3.2.1 Detailed Description	6
3.3 adc_seqX_ctrl Struct Reference	7
3.3.1 Detailed Description	7
3.4 adc_seqX_gdat Struct Reference	7
3.4.1 Detailed Description	8
3.5 adc_thr Struct Reference	8
3.5.1 Detailed Description	8
<b>4 File Documentation</b>	<b>9</b>
4.1 inc/ADC_FW.h File Reference	9
4.1.1 Detailed Description	11
4.1.2 Function Documentation	11
4.1.2.1 ADC_Disable()	11
4.1.2.2 ADC_Enable()	12
4.1.2.3 ADC_Init()	12
4.1.2.4 ADC_Power()	14
4.2 inc/Application.h File Reference	15
4.2.1 Detailed Description	15
4.2.2 Function Documentation	15
4.2.2.1 GPIO_Init()	16
4.2.2.2 LPC_Init()	16
4.3 inc/Disp7Seg_FW.h File Reference	17
4.3.1 Detailed Description	18
4.3.2 Function Documentation	18
4.3.2.1 DISP7SEG_Init()	18
4.3.2.2 DISP_Sweep()	19
4.4 inc/Disp7Seg_SW.h File Reference	20
4.4.1 Detailed Description	20
4.4.2 Function Documentation	21
4.4.2.1 Display()	21
4.5 inc/GPIO_FW.h File Reference	21
4.5.1 Detailed Description	24
4.5.2 Function Documentation	24

4.5.2.1	GetOFFSET()	24
4.5.2.2	GPIO_ClearOUT()	25
4.5.2.3	GPIO_Debounce()	25
4.5.2.4	GPIO_DebounceUserKEY()	26
4.5.2.5	GPIO_Disable()	27
4.5.2.6	GPIO_Enable()	27
4.5.2.7	GPIO_GetPIN()	28
4.5.2.8	GPIO_SetDIR()	29
4.5.2.9	GPIO_SetModeCLKDIV()	29
4.5.2.10	GPIO_SetModeDAC()	30
4.5.2.11	GPIO_SetModeFILTER()	31
4.5.2.12	GPIO_SetModeHYS()	32
4.5.2.13	GPIO_SetModeI2C()	32
4.5.2.14	GPIO_SetModeINPUT()	33
4.5.2.15	GPIO_SetModeINV()	34
4.5.2.16	GPIO_SetModeOD()	34
4.5.2.17	GPIO_SetOUT()	35
4.5.2.18	GPIO_SetPIN()	36
4.5.2.19	GPIO_ToogleOUT()	36
4.5.2.20	IOCONDisable()	37
4.5.2.21	IOCONEnable()	37
4.6	inc/GPIO_SW.h File Reference	38
4.6.1	Detailed Description	38
4.6.2	Function Documentation	38
4.6.2.1	GetInput()	39
4.6.2.2	GetUserKEY()	39
4.7	inc/LPC845.h File Reference	40
4.7.1	Detailed Description	40
4.8	inc/SwitchMatrix_FW.h File Reference	41
4.8.1	Detailed Description	42
4.8.2	Enumeration Type Documentation	42
4.8.2.1	anonymous enum	42
4.8.2.2	anonymous enum	42
4.8.2.3	anonymous enum	43
4.8.2.4	anonymous enum	43
4.8.3	Function Documentation	44
4.8.3.1	SWM()	44
4.8.3.2	SWM_Disable()	45
4.8.3.3	SWM_Enable()	45
4.8.3.4	SWM_PinEnable()	46
4.9	inc/SYSCON_FW.h File Reference	46
4.9.1	Detailed Description	49

4.9.2 Function Documentation	49
4.9.2.1 BoardClockRUN()	49
4.10 inc/SysTick_FW.h File Reference	50
4.10.1 Detailed Description	51
4.10.2 Function Documentation	51
4.10.2.1 SysTick_Init()	51
4.10.2.2 SysTick_Off()	52
4.10.2.3 SysTick_Set()	52
4.11 source/05-ConversionADC.c File Reference	53
4.11.1 Detailed Description	53
4.11.2 Function Documentation	54
4.11.2.1 main()	54
4.11.3 Variable Documentation	54
4.11.3.1 tick	55
4.12 source/ADC_FW.c File Reference	55
4.12.1 Detailed Description	55
4.12.2 Function Documentation	56
4.12.2.1 ADC0_SEQA_IRQHandler()	56
4.12.2.2 ADC_Disable()	57
4.12.2.3 ADC_Enable()	57
4.12.2.4 ADC_Init()	58
4.12.2.5 ADC_Power()	59
4.13 source/Aplication.c File Reference	60
4.13.1 Detailed Description	60
4.13.2 Function Documentation	61
4.13.2.1 GPIO_Init()	61
4.13.2.2 LPC_Init()	61
4.13.3 Variable Documentation	62
4.13.3.1 tick	62
4.14 source/Disp7Seg_FW.c File Reference	62
4.14.1 Detailed Description	63
4.14.2 Function Documentation	63
4.14.2.1 DISP7SEG_Init()	63
4.14.2.2 DISP_Sweep()	64
4.15 source/Disp7Seg_SW.c File Reference	65
4.15.1 Detailed Description	65
4.15.2 Function Documentation	66
4.15.2.1 Display()	66
4.15.3 Variable Documentation	66
4.15.3.1 Digits_to_BCD7seg	67
4.16 source/GPIO_FW.c File Reference	67
4.16.1 Detailed Description	68

4.16.2 Function Documentation	68
4.16.2.1 GetOFFSET()	68
4.16.2.2 GPIO_ClearOUT()	69
4.16.2.3 GPIO_Debounce()	70
4.16.2.4 GPIO_DebounceUserKEY()	70
4.16.2.5 GPIO_Disable()	71
4.16.2.6 GPIO_Enable()	72
4.16.2.7 GPIO_GetPIN()	72
4.16.2.8 GPIO_SetDIR()	73
4.16.2.9 GPIO_SetModeCLKDIV()	74
4.16.2.10 GPIO_SetModeDAC()	74
4.16.2.11 GPIO_SetModeFILTER()	75
4.16.2.12 GPIO_SetModeHYS()	76
4.16.2.13 GPIO_SetModeI2C()	76
4.16.2.14 GPIO_SetModeINPUT()	77
4.16.2.15 GPIO_SetModeINV()	78
4.16.2.16 GPIO_SetModeOD()	79
4.16.2.17 GPIO_SetOUT()	79
4.16.2.18 GPIO_SetPIN()	80
4.16.2.19 GPIO_ToogleOUT()	81
4.16.2.20 IOCONDisable()	81
4.16.2.21 IOCONEnable()	82
4.16.3 Variable Documentation	82
4.16.3.1 offset	83
4.17 source/GPIO_SW.c File Reference	83
4.17.1 Detailed Description	83
4.17.2 Function Documentation	83
4.17.2.1 GetInput()	84
4.17.2.2 GetUserKEY()	84
4.18 source/mtb.c File Reference	85
4.18.1 Detailed Description	85
4.19 source/SwitchMatrix_FW.c File Reference	86
4.19.1 Detailed Description	86
4.19.2 Function Documentation	86
4.19.2.1 SWM()	86
4.19.2.2 SWM_Disable()	87
4.19.2.3 SWM_Enable()	88
4.19.2.4 SWM_PinEnable()	88
4.20 source/SYSCON_FW.c File Reference	89
4.20.1 Detailed Description	89
4.20.2 Function Documentation	90
4.20.2.1 BoardClockRUN()	90

---

4.21 source/SysTick_FW.c File Reference . . . . .	90
4.21.1 Detailed Description . . . . .	91
4.21.2 Function Documentation . . . . .	91
4.21.2.1 SysTick_Handler() . . . . .	91
4.21.2.2 SysTick_Init() . . . . .	92
4.21.2.3 SysTick_Off() . . . . .	93
4.21.2.4 SysTick_Set() . . . . .	93
4.21.3 Variable Documentation . . . . .	94
4.21.3.1 tick . . . . .	94
<b>Index</b>	<b>95</b>





# Chapter 1

## Class Index

### 1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">adc_chan</a>	5
<a href="#">adc_ctrl_t</a>	6
<a href="#">adc_seqX_ctrl</a>	7
<a href="#">adc_seqX_gdat</a>	7
<a href="#">adc_thr</a>	8



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

inc/ <a href="#">ADC_FW.h</a>	: Firmware functions ADC	9
inc/ <a href="#">Aplication.h</a>	: Functions used in main	15
inc/ <a href="#">Disp7Seg_FW.h</a>	: Firmware functions for DISP7SEG	17
inc/ <a href="#">Disp7Seg_SW.h</a>	: Software functions for DISP7SEG	20
inc/ <a href="#">GPIO_FW.h</a>	: Firmware functions for GPIO	21
inc/ <a href="#">GPIO_SW.h</a>	: Software functions for GPIO	38
inc/ <a href="#">LPC845.h</a>	: Declarations for type of data	40
inc/ <a href="#">SwitchMatrix_FW.h</a>	: Firmware functions for SWM	41
inc/ <a href="#">SYSCON_FW.h</a>	: Firmware functions for SYSCON	46
inc/ <a href="#">SysTick_FW.h</a>	: Firmware functions for SysTick	50
source/ <a href="#">05-ConversionADC.c</a>	: Firmware functions ADC	53
source/ <a href="#">ADC_FW.c</a>	: Firmware functions ADC	55
source/ <a href="#">Aplication.c</a>	: Functions used in main	60
source/ <a href="#">Disp7Seg_FW.c</a>	: Firmware functions for DISP7SEG	62
source/ <a href="#">Disp7Seg_SW.c</a>	: Software functions for DISP7SEG	65
source/ <a href="#">GPIO_FW.c</a>	: Firmware functions for GPIO	67
source/ <a href="#">GPIO_SW.c</a>	: Software functions for GPIO	83
source/ <a href="#">mtb.c</a>	MTB initialization file	85

source/ <b>semihost_hardfault.c</b> . . . . .	??
source/ <a href="#">SwitchMatrix_FW.c</a> : Firmware functions for SWM . . . . .	86
source/ <a href="#">SYSCON_FW.c</a> : Firmware functions for SYSCON . . . . .	89
source/ <a href="#">SysTick_FW.c</a> : Firmware functions for SysTick . . . . .	90

## Chapter 3

# Class Documentation

### 3.1 adc\_chan Struct Reference

#### Public Attributes

- ```
union {
    __RW uint32_t _CHAN_THRSEL
    struct {
        __RW uint32_t _CH0_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH1_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH2_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH3_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH4_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH5_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH6_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH7_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH8_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH9_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH10_THRSEL:1
            Compare against THR.
        __RW uint32_t _CH11_THRSEL:1
            Compare against THR.
        __RW uint32_t _RESERVED:20
    }
};
```

### 3.1.1 Detailed Description

Definition at line 145 of file ADC\_FW.h.

The documentation for this struct was generated from the following file:

- inc/[ADC\\_FW.h](#)

## 3.2 adc\_ctrl\_t Struct Reference

### Public Attributes

- ```

union {
    __RW uint32_t \_CTRL
        < Union between _CTRL and bit field; They're overlapped
    struct {
        __RW uint32_t \_CLKDIV:8
            Clock divided by this + 1 to produce sampling clock <= 30MHz.
        __RW uint32_t \_ASYNCMODE:1
            Asynchronous operation mode.
        __RW uint32_t \_RESERVED\_0:1
        __RW uint32_t \_LPWRMODE:1
            Power down ADC while is not used.
        __RW uint32_t \_RESERVED\_1:19
        __RW uint32_t \_CALMODE:1
            Self calibration.
        __RW uint32_t \_RESERVED\_2:1
    }
};

```

< Struct for handling adc configuration

### 3.2.1 Detailed Description

Definition at line 56 of file ADC\_FW.h.

The documentation for this struct was generated from the following file:

- inc/[ADC\\_FW.h](#)

### 3.3 adc\_seqX\_ctrl Struct Reference

#### Public Attributes

- ```

union {
    __RW uint32_t _SEQx_CTRL
    struct {
        __RW uint32_t _CHANNELS:12
            Select which channel will be sampled.
        __RW uint32_t _TRIGGER:3
            Select which HW trigger will start conversion.
        __RW uint32_t _RESERVED_0:3
        __RW uint32_t _TRIGPOL:1
            Polarity of the input trigger.
        __RW uint32_t _SYNCBYPASS:1
            Bypass synchronization FF, so is slower.
        __RW uint32_t _TSAMP:5
        __RW uint32_t _RESERVED_1:1
        __RW uint32_t _START:1
            Launch one pass.
        __RW uint32_t _BURST:1
            Sequence continuously converted.
        __RW uint32_t _SINGLESTEP:1
            When start in 1 this converts only the next channel.
        __RW uint32_t _LOWPRIO:1
            Set priority for sequence A.
        __RW uint32_t _MODE:1
            Read global data or individual channel.
        __RW uint32_t _SEQx_ENA:1
            Enable sequence.
    }
};
      
```

#### 3.3.1 Detailed Description

Definition at line 74 of file ADC\_FW.h.

The documentation for this struct was generated from the following file:

- inc/ADC\_FW.h

### 3.4 adc\_seqX\_gdat Struct Reference

#### Public Attributes

-

```

union {
    __RW uint32_t _SEQx_GDAT
    struct {
        __RW uint32_t _RESERVED_0:4
        __RW uint32_t _RESULT:12
        12 bit A/D conversion
        __RW uint32_t _THCMPRANGE:2
        Compare the result with thrn_low and thrn_high.
        __RW uint32_t _THCMPCROSS:2
        Indicates a crossing of the threshold.
        __RW uint32_t _RESERVED_1:6
        __RW uint32_t _CHN:4
        Indicates the channel converted.
        __RW uint32_t _OVERRUN:1
        If a new conversion was loaded and the previous was not read.
        __RW uint32_t _DATAVALID:1
        There's a new result.
    }
};

```

### 3.4.1 Detailed Description

Definition at line 111 of file ADC\_FW.h.

The documentation for this struct was generated from the following file:

- inc/[ADC\\_FW.h](#)

## 3.5 adc\_thr Struct Reference

### Public Attributes

```

•
union {
    __RW uint32_t _THRn_LH
    struct {
        __RW uint32_t _RESERVED_0:4
        __RW uint32_t _THR:12
        12bits for compare
        __RW uint32_t _RESERVED_1:16
    }
};

```

### 3.5.1 Detailed Description

Definition at line 132 of file ADC\_FW.h.

The documentation for this struct was generated from the following file:

- inc/[ADC\\_FW.h](#)



## Chapter 4

# File Documentation

### 4.1 inc/ADC\_FW.h File Reference

: Firmware functions ADC

#### Classes

- struct [adc\\_ctrl\\_t](#)
- struct [adc\\_seqX\\_ctrl](#)
- struct [adc\\_seqX\\_gdat](#)
- struct [adc\\_thr](#)
- struct [adc\\_chan](#)

#### Macros

- #define **MASK\_ADC\_SYSCON** 4
- #define **ADC\_SYSAHB** 24
- #define **ADC\_0** PORT0,7,14  
*POT1 on board; 14 is the bit in PINENABLE.*
- #define **ADC\_ADD** ( ( \_\_RW uint32\_t \*) 0x4001C000UL)
- #define **\_ADC\_CTRL** ADC\_ADD[0]
- #define **\_ADC\_SEQA\_CTRL** ADC\_ADD[2]
- #define **\_ADC\_SEQB\_CTRL** ADC\_ADD[3]
- #define **\_ADC\_SEQA\_GDAT** ADC\_ADD[4]
- #define **\_ADC\_SEQB\_GDAT** ADC\_ADD[5]
- #define **\_ADC\_DAT0** ADC\_ADD[8]
- #define **\_ADC\_DAT1** ADC\_ADD[9]
- #define **\_ADC\_DAT2** ADC\_ADD[10]
- #define **\_ADC\_DAT3** ADC\_ADD[11]
- #define **\_ADC\_DAT4** ADC\_ADD[12]
- #define **\_ADC\_DAT5** ADC\_ADD[13]
- #define **\_ADC\_DAT6** ADC\_ADD[14]
- #define **\_ADC\_DAT7** ADC\_ADD[15]
- #define **\_ADC\_DAT8** ADC\_ADD[16]
- #define **\_ADC\_DAT9** ADC\_ADD[17]
- #define **\_ADC\_DAT10** ADC\_ADD[18]

- `#define _ADC_DAT11 ADC_ADD[19]`
- `#define _ADC_THR0_LOW ADC_ADD[20]`
- `#define _ADC_THR1_LOW ADC_ADD[21]`
- `#define _ADC_THR0_HIGH ADC_ADD[22]`
- `#define _ADC_THR1_HIGH ADC_ADD[23]`
- `#define _ADC_CHAN_THRSEL ADC_ADD[24]`
- `#define ADC_INTEN ADC_ADD[25]`
- `#define ADC_FLAGS ADC_ADD[26]`
- `#define ADC_TRM ADC_ADD[27]`
- `#define ADC_CTRL ( ( __RW adc_ctrl_t *) 0x4001C000UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_SEQA_CTRL ( ( __RW adc_seqX_ctrl *) 0x4001C008UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_SEQB_CTRL ( ( __RW adc_seqX_ctrl *) 0x4001C00CUL)`  
*Pointer to a struct in that memory.*
- `#define ADC_SEQA_GDAT ( ( __RW adc_seqX_gdat *) 0x4001C010UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_SEQB_GDAT ( ( __RW adc_seqX_gdat *) 0x4001C014UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT0 ( ( __RW adc_seqX_gdat *) 0x4001C020UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT1 ( ( __RW adc_seqX_gdat *) 0x4001C024UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT2 ( ( __RW adc_seqX_gdat *) 0x4001C028UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT3 ( ( __RW adc_seqX_gdat *) 0x4001C02CUL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT4 ( ( __RW adc_seqX_gdat *) 0x4001C030UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT5 ( ( __RW adc_seqX_gdat *) 0x4001C034UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT6 ( ( __RW adc_seqX_gdat *) 0x4001C038UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT7 ( ( __RW adc_seqX_gdat *) 0x4001C03CUL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT8 ( ( __RW adc_seqX_gdat *) 0x4001C040UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT9 ( ( __RW adc_seqX_gdat *) 0x4001C044UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT10 ( ( __RW adc_seqX_gdat *) 0x4001C048UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_DAT11 ( ( __RW adc_seqX_gdat *) 0x4001C04CUL)`  
*Pointer to a struct in that memory.*
- `#define ADC_THR0_LOW ( ( __RW adc_thr *) 0x4001C050UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_THR1_LOW ( ( __RW adc_thr *) 0x4001C054UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_THR0_HIGH ( ( __RW adc_thr *) 0x4001C058UL)`  
*Pointer to a struct in that memory.*
- `#define ADC_THR1_HIGH ( ( __RW adc_thr *) 0x4001C05CUL)`  
*Pointer to a struct in that memory.*

- #define `ADC_CHAN_THRSEL` ( ( \_\_RW `adc_chan` \*) 0x4001C060UL)  
*Pointer to a struct in that memory.*
- #define `MASK_SEQA_INTEN` 1<<0  
*Interrupt after each conv.*
- #define `MASK_SEQB_INTEN` 1<<1  
*Interrupt after each conv.*
- #define `MASK_ISE_ADC_SEQA` 1<<16  
*Enable Interrupt NVIC.*
- #define `MASK_ISE_ADC_SEQB` 1<<17  
*Enable Interrupt NVIC.*

## Functions

- void `ADC_Init` (uint8\_t port, uint8\_t pin, uint8\_t ena)  
: Initialize ADC on a pin
- void `ADC_Power` (void)  
: Power ADC
- void `ADC_Enable` (void)  
: Enable clock in ADC
- void `ADC_Disable` (void)  
: Disable clock in ADC

### 4.1.1 Detailed Description

: Firmware functions ADC

: 12 bits conversion

Author

: Tobias Bavasso Piizzi

Date

: 08/01/2021

### 4.1.2 Function Documentation

#### 4.1.2.1 `ADC_Disable()`

```
void ADC_Disable (
    void )
```

: Disable clock in ADC

:

Author

: Tobias Bavasso Piizzi

Date

: 08/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 90 of file ADC\_FW.c.

```

90         {
91     SYSAHBCLKCTRL0&= (~ (1<<ADC_SYSAHB));
92 }
```

**4.1.2.2 ADC\_Enable()**

```

void ADC_Enable (
    void )
```

: Enable clock in ADC

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 08/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 77 of file ADC\_FW.c.

```

77         {
78     SYSAHBCLKCTRL0|= (1<<ADC_SYSAHB);
79 }
```

**4.1.2.3 ADC\_Init()**

```

void ADC_Init (
    uint8_t port,
```

```
uint8_t pin,  
uint8_t ena )
```

: Initialize ADC on a pin

: Continuous conversion of POTE in board

#### Author

: Tobias Bavasso Piizzi

#### Date

: 08/01/2021

#### Parameters

|  |                                                           |
|--|-----------------------------------------------------------|
|  | [in] uint8_t port: PORT0,PORT1                            |
|  | [in] uint8_t pin: 0,31                                    |
|  | [in] uint8_t en: bit to enable in PINENABLE (page 143 UM) |

#### Returns

: void

< Enable CLOCK in SYSAHB

< Enable service interrupt

< Interrupt after conversion finish

< Enable Switch Matrix

< Enable pin in SWN as AnalogInput

< Disable Switch Matrix

< Power in SYSCON

< Div = 0

< Sync

< OFF

< OFF

< Sample CH0

< No hardware trigger

< Positive trigger

< Enable sync

< Individual end of conversion

< Start,enable set on the same line first time

Definition at line 23 of file ADC\_FW.c.

```

23                                     {
24
25     ADC_Enable();
26     ISER0|= MASK_ISE_ADC_SEQA;
27     ADC_INTEN|= MASK_SEQA_INTEN;
28     SWM_Enable();
29     SWM_PinEnable(port, pin, ena);
30     SWM_Disable();
31     ADC_Power();
32
33
34
35     ADC_CTRL->_CLKDIV = 0x00;
36     ADC_CTRL->_ASYNCMODE = 0;
37     ADC_CTRL->_LPWRMODE = 0;
38     ADC_CTRL->_CALMODE = 0;
39
40     ADC_SEQA_CTRL->_CHANNELS      = 0x01;
41     ADC_SEQA_CTRL->_TRIGGER       = 0x00;
42     ADC_SEQA_CTRL->_TRIGPOL      = 0x1;
43     ADC_SEQA_CTRL->_SYNCBYPASS    = 0x0;
44     ADC_SEQA_CTRL->_TSAMP        = 0x00;
45     ADC_SEQA_CTRL->_START        = 0;
46     ADC_SEQA_CTRL->_BURST        = 0;
47     ADC_SEQA_CTRL->_SINGLESTEP     = 0x0;
48     ADC_SEQA_CTRL->_LOWPRIO      = 0x0;
49     ADC_SEQA_CTRL->_MODE         = 0;
50     ADC_SEQA_CTRL->_SEQx_ENA     = 0;
51     _ADC_SEQA_CTRL |= ((0b100001) << 26);
52 }
```

#### 4.1.2.4 ADC\_Power()

```

void ADC_Power (
    void )
```

: Power ADC

:

Author

: Tobias Bavasso Piizzi

Date

: 08/01/2021

Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

Returns

: void

Definition at line 63 of file ADC\_FW.c.

```
63         {
64     PDRUNCFG&= (~ (1 « MASK_ADC_SYSCON));
65
66 }
```

## 4.2 inc/Application.h File Reference

: Functions used in main

```
#include "LPC845.h"
#include "GPIO_FW.h"
#include "GPIO_SW.h"
#include "SwitchMatrix_FW.h"
#include "SYSCON_FW.h"
#include "SysTick_FW.h"
#include "Disp7Seg_FW.h"
#include "Disp7Seg_SW.h"
#include "ADC_FW.h"
```

### Functions

- void [LPC\\_Init](#) (void)  
: *Initialize the board*
- void [GPIO\\_Init](#) (void)  
: *Initialize the GPIO*

### 4.2.1 Detailed Description

: Functions used in main

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

### 4.2.2 Function Documentation

#### 4.2.2.1 GPIO\_Init()

```
void GPIO_Init (
    void )
```

: Initialize the GPIO

: It depends on each project

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|           |
|-----------|
| [in] void |
|-----------|

##### Returns

: void

Definition at line 35 of file Application.c.

```
35     {
36         GPIO_SetDIR(UserKEY, INPUT);
37         GPIO_SetDIR(LedGREEN, OUTPUT);
38         GPIO_SetDIR(LedBLUE, OUTPUT);
39
40         GPIO_SetPIN(LedGREEN, LED_OFF);
41         GPIO_SetPIN(LedBLUE, LED_OFF);
42     }
```

#### 4.2.2.2 LPC\_Init()

```
void LPC_Init (
    void )
```

: Initialize the board

: It depends on each project

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021



## Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

## Returns

: void

Definition at line 19 of file Application.c.

```

19      {
20          GPIO_Enable();
21          BoardClockRUN();
22          SysTick_Init();
23          GPIO_Init();
24      }
```

## 4.3 inc/Disp7Seg\_FW.h File Reference

: Firmware functions for DISP7SEG

### Macros

- #define [SEG\\_A](#) PORT0,21  
*Pin to connect segA.*
- #define [SEG\\_B](#) PORT0,22  
*Pin to connect segB.*
- #define [SEG\\_C](#) PORT0,16  
*Pin to connect segC.*
- #define [SEG\\_D](#) PORT0,17  
*Pin to connect segD.*
- #define [SEG\\_E](#) PORT0,18  
*Pin to connect segE.*
- #define [SEG\\_F](#) PORT0,20  
*Pin to connect segF.*
- #define [SEG\\_G](#) PORT0,19  
*Pin to connect segG.*
- #define [SEG\\_DP](#) PORT0,23  
*Pin to connect segDP.*
- #define [TR\\_D1](#) PORT0,0  
*Pin to connect transistor DISP1.*
- #define [TR\\_D0](#) PORT0,1  
*Pin to connect transistor DISP0.*
- #define [DIGITS](#) 2  
*Number of displays.*
- #define [DIGIT\\_0](#) 0
- #define [DIGIT\\_1](#) 1

## Functions

- void `DISP7SEG_Init` (void)  
: Set pins for display as out
- void `DISP_Sweep` (void)  
: Refresh the display 7Seg (2 Disp)

### 4.3.1 Detailed Description

: Firmware functions for DISP7SEG

:

Author

: Tobias Bavasso Piizzi

Date

: 07/01/2021

### 4.3.2 Function Documentation

#### 4.3.2.1 DISP7SEG\_Init()

```
void DISP7SEG_Init (  
    void )
```

: Set pins for display as out

:

Author

: Tobias Bavasso Piizzi

Date

: 07/01/2021

Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 19 of file Disp7Seg\_FW.c.

```

19      {
20          GPIO_SetDIR(SEG_A, OUTPUT);
21          GPIO_SetDIR(SEG_B, OUTPUT);
22          GPIO_SetDIR(SEG_C, OUTPUT);
23          GPIO_SetDIR(SEG_D, OUTPUT);
24          GPIO_SetDIR(SEG_E, OUTPUT);
25          GPIO_SetDIR(SEG_F, OUTPUT);
26          GPIO_SetDIR(SEG_G, OUTPUT);
27          GPIO_SetDIR(TR_D0, OUTPUT);
28          GPIO_SetDIR(TR_D1, OUTPUT);
29
30          GPIO_ClearOUT(SEG_A);
31          GPIO_ClearOUT(SEG_B);
32          GPIO_ClearOUT(SEG_C);
33          GPIO_ClearOUT(SEG_D);
34          GPIO_ClearOUT(SEG_E);
35          GPIO_ClearOUT(SEG_F);
36          GPIO_ClearOUT(SEG_G);
37          GPIO_ClearOUT(TR_D0);
38          GPIO_ClearOUT(TR_D1);
39      }

```

**4.3.2.2 DISP\_Sweep()**

```

void DISP_Sweep (
    void )

```

: Refresh the display 7Seg (2 Disp)

: Is necessary to be used in SysTick\_Handler

**Author**

: Tobias Bavasso Piizzi

**Date**

: 07/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

&lt; Number of disp

&lt; Turn off transistor

&lt; Turn off transistor

< Next time sweep other disp

< Reset the digits

Definition at line 51 of file Disp7Seg\_FW.c.

```

51     {
52         uint8_t aux;
53         static uint8_t digit = 0;
54
55         GPIO_ClearOUT(TR_D0);
56         GPIO_ClearOUT(TR_D1);
57
58         aux = buff_Dis7[digit];
59
60         GPIO_SetPIN( SEG_A, ((aux >> 0) & (uint8_t) 0x01));
61         GPIO_SetPIN( SEG_B, ((aux >> 1) & (uint8_t) 0x01));
62         GPIO_SetPIN( SEG_C, ((aux >> 2) & (uint8_t) 0x01));
63         GPIO_SetPIN( SEG_D, ((aux >> 3) & (uint8_t) 0x01));
64         GPIO_SetPIN( SEG_E, ((aux >> 4) & (uint8_t) 0x01));
65         GPIO_SetPIN( SEG_F, ((aux >> 5) & (uint8_t) 0x01));
66         GPIO_SetPIN( SEG_G, ((aux >> 6) & (uint8_t) 0x01));
67         GPIO_SetPIN( SEG_DP, ((aux >> 7) & (uint8_t) 0x01));
68
69         switch (digit) {
70             case DIGIT_0:
71                 GPIO_SetOUT(TR_D0);
72                 break;
73             case DIGIT_1:
74                 GPIO_SetOUT(TR_D1);
75                 break;
76             default:
77                 digit = 0;
78                 GPIO_SetOUT(TR_D0);
79                 break;
80         }
81
82         digit++;
83         digit %= DIGITS;
84
85     }

```

## 4.4 inc/Disp7Seg\_SW.h File Reference

: Software functions for DISP7SEG

### Functions

- void [Display](#) (uint8\_t val)  
: Writes on Disp7Seg

#### 4.4.1 Detailed Description

: Software functions for DISP7SEG

:

Author

: Tobias Bavasso Piizzi

Date

: 07/01/2021

## 4.4.2 Function Documentation

### 4.4.2.1 Display()

```
void Display (
    uint8_t val )
```

: Writes on Disp7Seg

: High lever of layers

#### Author

: Tobias Bavasso Piizzi

#### Date

: 07/01/2021

#### Parameters

|                           |
|---------------------------|
| [in] uint8_t val: 0 to 99 |
|---------------------------|

#### Returns

: void

< Disable SysTick INT

< Enable SysTick INT

Definition at line 38 of file Disp7Seg\_SW.c.

```
38     {
39         uint8_t i;
40         uint8_t auxDisp[DIGITS];
41
42         for (i = 0; i < DIGITS; i++) {
43             auxDisp[i] = Digits_to_BCD7seg[val % 10];
44             val /= 10;
45         }
46         for (i = 0; i < DIGITS; i++) {
47             SYSTICK_INT_DIS;
48             buff_Dis7[i] = auxDisp[i];
49             SYSTICK_INT_EN;
50         }
51     }
52 }
```

## 4.5 inc/GPIO\_FW.h File Reference

: Firmware functions for GPIO

## Macros

- #define **PORT0** 0
- #define **PORT1** 1
- #define **LedGREEN** PORT1 , 0  
*Led green in board.*
- #define **LedBLUE** PORT1 , 1  
*Led blue in board.*
- #define **LedRED** PORT1 , 2  
*Led red in board.*
- #define **UserKEY** PORT0 , 4  
*Key in board.*
- #define **INPUT** 0
- #define **OUTPUT** 1
- #define **LOW** 0
- #define **HIGH** 1
- #define **ACT\_HIGH** 1
- #define **ACT\_LOW** 0
- #define **LED\_ON** 0  
*The led are active low.*
- #define **LED\_OFF** 1  
*The led are active low.*
- #define **BOUNCE** 10  
*Times to check the bounce.*
- #define **SysAHBCLKCTRL** ( ( \_\_RW uint32\_t \*) 0x40048080UL)
- #define **SysAHBCLKCTRL0** SysAHBCLKCTRL[0]
- #define **SysAHBCLKCTRL1** SysAHBCLKCTRL[1]
- #define **GPIO\_PBYTE** ( ( \_\_RW uint8\_t \*) 0xA0000000UL)
- #define **GPIO\_PWORD** ( ( \_\_RW uint32\_t \*) 0xA0001000UL)
- #define **GPIO\_DIRP** ( ( \_\_RW uint32\_t \*) 0xA0002000UL)
- #define **GPIO\_PORT** ( ( \_\_RW uint32\_t \*) 0xA0002100UL)
- #define **GPIO\_SETP** ( ( \_\_RW uint32\_t \*) 0xA0002200UL)
- #define **GPIO\_CLRP** ( ( \_\_RW uint32\_t \*) 0xA0002280UL)
- #define **GPIO\_NOTP** ( ( \_\_RW uint32\_t \*) 0xA0002300UL)
- #define **NO\_PULL\_UP\_DOWN** 0x00
- #define **PULL\_DOWN** 0x01
- #define **PULL\_UP** 0x02
- #define **REPEATER** 0x03
- #define **HYS\_EN** 0x01
- #define **HYS\_DIS** 0x00
- #define **INV\_INPUT** 0x01
- #define **NOT\_INV\_INPUT** 0x00
- #define **OD\_EN** 0x01
- #define **OD\_DIS** 0x00
- #define **BYPASS\_FILTER** 0x00
- #define **CLK1\_FILTER** 0x01
- #define **CLK2\_FILTER** 0x02
- #define **CLK3\_FILTER** 0x03
- #define **IOCONCLKDIV0** 0x00
- #define **IOCONCLKDIV1** 0x01
- #define **IOCONCLKDIV2** 0x02
- #define **IOCONCLKDIV3** 0x03
- #define **IOCONCLKDIV4** 0x04
- #define **IOCONCLKDIV5** 0x05

- #define **IOCONCLKDIV6** 0x06
- #define **DAC\_EN** 0x01
- #define **DAC\_DIS** 0x00
- #define **STD\_MODE** 0x00
- #define **STD\_GPIO** 0x01
- #define **FAST\_MODE** 0x02
- #define **IOCON\_** ( ( \_\_RW uint32\_t \*) 0x40044000UL)

## Functions

- void **GPIO\_Enable** (void)  
: Enable GPIO0 and GPIO1
- void **GPIO\_Disable** (void)  
: Disable GPIO0 and GPIO1
- void **GPIO\_SetDIR** (uint8\_t port, uint8\_t pin, uint8\_t dir)  
: Choose GPIO as Input/Output
- void **GPIO\_SetPIN** (uint8\_t port, uint8\_t pin, uint8\_t state)  
: Choose GPIO's output state
- uint8\_t **GPIO\_GetPIN** (uint8\_t port, uint8\_t pin, uint8\_t state)  
: Return GPIO's input state
- void **GPIO\_SetOUT** (uint8\_t port, uint8\_t pin)  
: Put GPIO's out to 1
- void **GPIO\_ClearOUT** (uint8\_t port, uint8\_t pin)  
: Put GPIO's out to 0
- void **GPIO\_ToggleOUT** (uint8\_t port, uint8\_t pin)  
: Invert GPIO's out
- void **GPIO\_DebounceUserKEY** (void)  
: Firmware debounce for user key in board
- void **GPIO\_Debounce** (uint8\_t port, uint8\_t pin, uint8\_t state)  
: Firmware debounce for a GPIO
- void **IOCONEnable** (void)  
: Enable IOCON
- void **IOCONDisable** (void)  
: Disable IOCON
- uint8\_t **GetOFFSET** (uint8\_t port, uint8\_t pin)  
: Usefull for SetMode functions
- void **GPIO\_SetModeINPUT** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: on-chip pull-up/pull-down resistor
- void **GPIO\_SetModeHYS** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: Hysteresis
- void **GPIO\_SetModeINV** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: Invert input
- void **GPIO\_SetModeOD** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: Open drain
- void **GPIO\_SetModeFILTER** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: Digital filter sample mode
- void **GPIO\_SetModeCLKDIV** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: Select peripheral clock divider for input filter sampling clock
- void **GPIO\_SetModeDAC** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: Selects DAC mode
- void **GPIO\_SetModeI2C** (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: Selects I2C mode

### 4.5.1 Detailed Description

: Firmware functions for GPIO

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

### 4.5.2 Function Documentation

#### 4.5.2.1 GetOFFSET()

```
uint8_t GetOFFSET (  
    uint8_t port,  
    uint8_t pin )
```

: Usefull for SetMode functions

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|                                 |
|---------------------------------|
| [in] uint8_t port : PORT0,PORT1 |
| [in] uint8_t pin : 0,31         |

#### Returns

: void

Definition at line 231 of file GPIO\_FW.c.

```
231  
232     uint8_t index;  
233     index = port * 32 + pin;  
234     return ((offset[index]) / 4);  
235 }
```



#### 4.5.2.2 GPIO\_ClearOUT()

```
void GPIO_ClearOUT (
    uint8_t port,
    uint8_t pin )
```

: Put GPIO's out to 0

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|                   |               |
|-------------------|---------------|
| [in] uint8_t port | : PORT0,PORT1 |
| [in] uint8_t pin  | : 0,31        |

##### Returns

: void

Definition at line 113 of file GPIO\_FW.c.

```
113 {
114     GPIO_CLR_P[port] |= (1 << pin);
115 }
```

#### 4.5.2.3 GPIO\_Debounce()

```
void GPIO_Debounce (
    uint8_t port,
    uint8_t pin,
    uint8_t state )
```

: Firmware debounce for a GPIO

: Use in SysTick\_Handler or in some timer interrupt

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

**Parameters**

|  |                                       |
|--|---------------------------------------|
|  | [in] uint8_t port : PORT0,PORT1       |
|  | [in] uint8_t pin : 0,31               |
|  | [in] uint8_t state : ACT_LOW,ACT_HIGH |

**Returns**

: void

Definition at line 169 of file GPIO\_FW.c.

```

169                                     {
170     static uint8_t q = 0;    //Quantity of bounces
171     uint8_t j = 0;          //It captures changes
172
173     if (GPIO_GetPIN(port, pin, state))    // The key is pushed?
174         j = 0x01;                        //Something is happening, the key is been pushed
175
176     if (buff_In ^ j) {                // If the key is pushed while q != BOUNCE
177         q++;                          // I change the buffer
178         if (q == BOUNCE) {
179             q = 0;
180             buff_In ^= 0x01;
181         }
182     } else
183         q = 0;
184 }

```

**4.5.2.4 GPIO\_DebounceUserKEY()**

```

void GPIO_DebounceUserKEY (
    void )

```

: Firmware debounce for user key in board

: Use in SysTick\_Handler or in some timer interrupt

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |      |
|--|------|
|  | [in] |
|--|------|

**Returns**

: void

Definition at line 141 of file GPIO\_FW.c.

```

141     {
142     static uint8_t q = 0;    //Quantity of bounces
143     uint8_t j = 0;         //It captures changes
144
145     if (GPIO_GetPIN(UserKEY, ACT_LOW))    // The key is pushed?
146         j = 0x01;                        //Something is happening, the key is been pushed
147
148     if (buff_UserKEY ^ j) {                // If the key is pushed while q != BOUNCE
149         q++;                               // I change the buffer
150         if (q == BOUNCE) {
151             q = 0;
152             buff_UserKEY ^= 0x01;
153         }
154     } else
155         q = 0;
156 }

```

#### 4.5.2.5 GPIO\_Disable()

```

void GPIO_Disable (
    void )

```

: Disable GPIO0 and GPIO1

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: void

Definition at line 32 of file GPIO\_FW.c.

```

32     {
33     SYSAHBCLKCTRL0 &= (~ (1<<6));
34     SYSAHBCLKCTRL0 &= (~ (1<<20));
35 }

```

#### 4.5.2.6 GPIO\_Enable()

```

void GPIO_Enable (
    void )

```

: Enable GPIO0 and GPIO1

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 19 of file GPIO\_FW.c.

```
19      {
20          SYSAHBCLKCTRL0 |= (1<<6);
21          SYSAHBCLKCTRL0 |= (1<<20);
22      }
```

**4.5.2.7 GPIO\_GetPIN()**

```
uint8_t GPIO_GetPIN (
    uint8_t port,
    uint8_t pin,
    uint8_t state )
```

: Return GPIO's input state

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |                                       |
|--|---------------------------------------|
|  | [in] uint8_t port : PORT0,PORT1       |
|  | [in] uint8_t pin : 0,31               |
|  | [in] uint8_t STATE : ACT_LOW,ACT_HIGH |

**Returns**

: uint8\_t : 1 pin == [state] , 0 pin != [state]

Definition at line 81 of file GPIO\_FW.c.

```

81                                     {
82     port = port * 32 + pin;
83     if ( GPIO_PBYTE[port] == state)
84         return 1;
85     else
86         return 0;
87 }
```

**4.5.2.8 GPIO\_SetDIR()**

```

void GPIO_SetDIR (
    uint8_t port,
    uint8_t pin,
    uint8_t dir )
```

: Choose GPIO as Input/Output

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |
|  | [in] uint8_t dir : INPUT,OUTPUT |

**Returns**

: void

Definition at line 48 of file GPIO\_FW.c.

```

48                                     {
49     GPIO_DIRP[port] &= (~(1 << pin));
50     GPIO_DIRP[port] |= (dir << pin);
51 }
```

**4.5.2.9 GPIO\_SetModeCLKDIV()**

```

void GPIO_SetModeCLKDIV (
    uint8_t port,
```

```
uint8_t pin,
uint8_t mode )
```

: Select peripheral clock divider for input filter sampling clock

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|                                                                                                           |
|-----------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: IOCONCLKDIV0 to IOCONCLKDIV6 |
|-----------------------------------------------------------------------------------------------------------|

#### Returns

: void

Definition at line 338 of file GPIO\_FW.c.

```
338
339     uint8_t offset;
340     offset = GetOFFSET(port, pin);
341     IOCON_[offset] &= (~ (0x07 « 13));
342     IOCON_[offset] |= (mode « 13);
343 }
```

#### 4.5.2.10 GPIO\_SetModeDAC()

```
void GPIO_SetModeDAC (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Selects DAC mode

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

## Parameters

|                                                                                             |
|---------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: DAC_EN,DAC_DIS |
|---------------------------------------------------------------------------------------------|

## Returns

: void

Definition at line 356 of file GPIO\_FW.c.

```

356                                     {
357     uint8_t offset;
358     offset = GetOFFSET(port, pin);
359     IOCON_[offset] &= (~(0x01 « 16));
360     IOCON_[offset] |= (mode « 16);
361 }
```

## 4.5.2.11 GPIO\_SetModeFILTER()

```

void GPIO_SetModeFILTER (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Digital filter sample mode

:

## Author

: Tobias Bavasso Piizzi

## Date

: 04/01/2021

## Parameters

|                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: BYPASS_FILTER,CLK1_FILTER,CLK2_FILTER,CLK3_FILTER |
|--------------------------------------------------------------------------------------------------------------------------------|

## Returns

: void

Definition at line 320 of file GPIO\_FW.c.

```

320                                     {
321     uint8_t offset;
322     offset = GetOFFSET(port, pin);
323     IOCON_[offset] &= (~(0x03 « 11));
324     IOCON_[offset] |= (mode « 11);
325 }
```

#### 4.5.2.12 GPIO\_SetModeHYS()

```
void GPIO_SetModeHYS (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Hysteresis

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |                                                                                            |
|--|--------------------------------------------------------------------------------------------|
|  | [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode:HYS_EN,HYS_DIS |
|--|--------------------------------------------------------------------------------------------|

##### Returns

: void

Definition at line 266 of file GPIO\_FW.c.

```
266                                     {
267     uint8_t offset;
268     offset = GetOFFSET(port, pin);
269     IOCON_[offset] &= (~ (0x01 « 5));
270     IOCON_[offset] |= (mode « 5);
271 }
```

#### 4.5.2.13 GPIO\_SetModeI2C()

```
void GPIO_SetModeI2C (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Selects I2C mode

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021



## Parameters

|                                                                                                         |
|---------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode:STD_MODE,STD_GPIO,FAST_MODE |
|---------------------------------------------------------------------------------------------------------|

## Returns

: void

Definition at line 374 of file GPIO\_FW.c.

```

374                                     {
375     uint8_t offset;
376     offset = GetOFFSET(port, pin);
377     IOCON_[offset] &= (~ (0x03 « 8));
378     IOCON_[offset] |= (mode « 8);
379 }
```

## 4.5.2.14 GPIO\_SetModeINPUT()

```

void GPIO_SetModeINPUT (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: on-chip pull-up/pull-down resistor

:

## Author

: Tobias Bavasso Piizzi

## Date

: 04/01/2021

## Parameters

|                                                                                                                        |
|------------------------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode:NO_PULL_UP_DOWN,PULL_DOWN,PULL_UP,REPEATER |
|------------------------------------------------------------------------------------------------------------------------|

## Returns

: void

Definition at line 248 of file GPIO\_FW.c.

```

248                                     {
249     uint8_t offset;
250     offset = GetOFFSET(port, pin);
251     IOCON_[offset] &= (~ (0x03 « 3));
252     IOCON_[offset] |= (mode « 3);
253 }
```

#### 4.5.2.15 GPIO\_SetModeINV()

```
void GPIO_SetModeINV (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Invert input

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |                                                                                                      |
|--|------------------------------------------------------------------------------------------------------|
|  | [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: INV_INPUT,NOT_INV_INPUT |
|--|------------------------------------------------------------------------------------------------------|

##### Returns

: void

Definition at line 284 of file GPIO\_FW.c.

```
284                                     {
285     uint8_t offset;
286     offset = GetOFFSET(port, pin);
287     IOCON[offset] &= (~ (0x01 « 6));
288     IOCON[offset] |= (mode « 6);
289 }
```

#### 4.5.2.16 GPIO\_SetModeOD()

```
void GPIO_SetModeOD (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Open drain

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

**Parameters**

|  |                                                                                           |
|--|-------------------------------------------------------------------------------------------|
|  | [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: OD_EN,OD_DIS |
|--|-------------------------------------------------------------------------------------------|

**Returns**

: void

Definition at line 302 of file GPIO\_FW.c.

```

302                                     {
303     uint8_t offset;
304     offset = GetOFFSET(port, pin);
305     IOCON_[offset] &= (~(0x01 « 10));
306     IOCON_[offset] |= (mode « 10);
307 }
```

**4.5.2.17 GPIO\_SetOUT()**

```

void GPIO_SetOUT (
    uint8_t port,
    uint8_t pin )
```

: Put GPIO's out to 1

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |

**Returns**

: void

Definition at line 99 of file GPIO\_FW.c.

```

99                                     {
100     GPIO_SETP[port] |= (1 « pin);
101 }
```

#### 4.5.2.18 GPIO\_SetPIN()

```
void GPIO_SetPIN (
    uint8_t port,
    uint8_t pin,
    uint8_t state )
```

: Choose GPIO's output state

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |
|  | [in] uint8_t state : LOW,HIGH   |

##### Returns

: void

Definition at line 64 of file GPIO\_FW.c.

```
64                                     {
65     port = port * 32 + pin;
66     GPIO_PBYTE[port] &= (~1);
67     GPIO_PBYTE[port] |= state;
68 }
```

#### 4.5.2.19 GPIO\_ToogleOUT()

```
void GPIO_ToogleOUT (
    uint8_t port,
    uint8_t pin )
```

: Invert GPIO's out

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

**Parameters**

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |

**Returns**

: void

Definition at line 127 of file GPIO\_FW.c.

```

127
128     GPIO_NOTP[port] |= (1 << pin);
129 }
```

**4.5.2.20 IOCONDisable()**

```

void IOCONDisable (
    void )
```

: Disable IOCON

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |      |
|--|------|
|  | [in] |
|--|------|

**Returns**

: void

Definition at line 208 of file GPIO\_FW.c.

```

208     {
209     SYSAHBCLKCTRL0&= (~ (1<<18));
210 }
```

**4.5.2.21 IOCONEnable()**

```

void IOCONEnable (
    void )
```

: Enable IOCON

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |      |
|--|------|
|  | [in] |
|--|------|

**Returns**

: void

Definition at line 195 of file GPIO\_FW.c.

```

195      {
196          SYSAHBCLKCTRL0 |= (1<<18);
197      }

```

## 4.6 inc/GPIO\_SW.h File Reference

: Software functions for GPIO

### Functions

- uint8\_t [GetUserKEY](#) (void)  
: State of the user key in board
- uint8\_t [GetInput](#) (void)  
: State of the input

#### 4.6.1 Detailed Description

: Software functions for GPIO

: These are functions in a higher layer of abstraction

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

#### 4.6.2 Function Documentation

#### 4.6.2.1 GetInput()

```
uint8_t GetInput (
    void )
```

: State of the input

: Is necessary using GPIO\_Debounce

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: uint8\_t 1 if input pressed, 0 if input pressed

Definition at line 48 of file GPIO\_SW.c.

```
48     {
49         static uint8_t buff_before = 0x00;
50
51         if ( buff_In == 0x01 && buff_before == 0x00 ){
52             buff_before = 0x01;
53             return (1);
54         }
55         else if ( buff_In == 0x01 && buff_before == 0x01 )
56             return (0);
57         else if ( buff_In == 0x00 && buff_before == 0x01 )
58             return (0);
59         else
60             return (0);
61 }
```

#### 4.6.2.2 GetUserKEY()

```
uint8_t GetUserKEY (
    void )
```

: State of the user key in board

: Is necessary using GPIO\_DebounceUserKEY

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: uint8\_t 1 if user key pressed, 0 if user key not

Definition at line 21 of file GPIO\_SW.c.

```

21     {
22         static uint8_t buff_before = 0x00;
23
24         if ( buff_UserKEY == 0x01 && buff_before == 0x00 ){
25             buff_before = 0x01;
26             return (1);
27         }
28         else if ( buff_UserKEY == 0x01 && buff_before == 0x01 )
29             return (0);
30         else if ( buff_UserKEY == 0x00 && buff_before == 0x01 ){
31             buff_before = 0x00;
32             return (0);
33         }
34         else
35             return (0);
36     }

```

## 4.7 inc/LPC845.h File Reference

: Declarations for type of data

**Macros**

- #define **\_\_R** volatile const
- #define **\_\_W** volatile
- #define **\_\_RW** volatile
- #define **\_ISER** ( (\_\_RW uint32\_t \*) 0xE000E100UL)
- #define **ISER0** \_ISER[0]

**Typedefs**

- typedef unsigned int **uint32\_t**
- typedef unsigned short **uint16\_t**
- typedef unsigned char **uint8\_t**

### 4.7.1 Detailed Description

: Declarations for type of data

: Only contains macros

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021



## 4.8 inc/SwitchMatrix\_FW.h File Reference

: Firmware functions for SWM

### Macros

- `#define PINASSIGN ( (__RW uint32_t *) 0x4000C000UL)`
- `#define PINENABLE ( (__RW uint32_t *) 0x4000C1C0UL)`

### Enumerations

- enum { **BYTE0**, **BYTE1**, **BYTE2**, **BYTE3** }
- enum {  
**UO\_TXD**, **UO\_SCLK**, **U1\_CTS**, **U2\_RTS**,  
**SPI0\_MOSI**, **SPI0\_SSEL2**, **SPI1\_MISO**, **SCT\_IN1**,  
**SCT\_OUT1**, **SCT\_OUT5**, **I2C2\_SDA**, **COMP0\_OUT**,  
**UART3\_RXD**, **UART4\_SCLK**, **T0\_MAT3** }
- enum {  
**U0\_RXD**, **U1\_TXD**, **U0\_SCLK**, **U2\_CTS**,  
**SPI0\_MISO**, **SPI0\_SSEL3**, **SPI1\_SSEL0**, **SCT\_IN2**,  
**SCT\_OUT2**, **SCT\_OUT6**, **I2C2\_SCL**, **CLKOUT**,  
**UART3\_SCLK**, **T0\_MAT0**, **T0\_CAP0** }
- enum {  
**UO\_RTS**, **U1\_RXD**, **U2\_TXD**, **U2\_SCLK**,  
**SPI0\_SSEL0**, **SPI1\_SCK**, **SPI1\_SSEL1**, **SCT\_IN3**,  
**SCT\_OUT3**, **I2C1\_SDA**, **I2C3\_SDA**, **GPIO\_INT\_BMAT**,  
**UART4\_TXD**, **T0\_MAT1**, **T0\_CAP1** }
- enum {  
**UO\_CTS**, **U1\_RTS**, **UO\_RXD**, **SPIO\_SCK**,  
**SPI0\_SSEL1**, **SPI1\_MOSI**, **SCT0\_IN0**, **SCT\_OUT0**,  
**SCT\_OUT4**, **I2C1\_SCL**, **I2C3\_SCL**, **UART3\_TXD**,  
**UART4\_RXD**, **T0\_MAT2**, **T0\_CAP2** }
- enum {  
**ADC\_0**, **ADC\_1**, **ADC\_2**, **ADC\_3**,  
**ADC\_4**, **ADC\_5**, **ADC\_6**, **ADC\_7**,  
**ADC\_8**, **ADC\_9**, **ADC\_10**, **ADC\_11**,  
**DACOUT0**, **DACOUT1**, **CAPT\_X0**, **CAPT\_X1**,  
**CAPT\_X2**, **CAPT\_X3** }
- enum {  
**CAPT\_X4**, **CAPT\_X5**, **CAPT\_X6**, **CAPT\_X7**,  
**CAPT\_X8**, **CAPT\_YL**, **CAPT\_YH** }

### Functions

- void **SWM** (uint8\_t port, uint8\_t pin, uint8\_t assign, uint8\_t byte)  
: Assign movable functions for pin
- void **SWM\_PinEnable** (uint8\_t port, uint8\_t pin, uint8\_t ena)  
: Enable pin works as value passed in ena
- void **SWM\_Enable** (void)  
: Enable SWM
- void **SWM\_Disable** (void)  
: Disable SWM

### 4.8.1 Detailed Description

: Firmware functions for SWM

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

### 4.8.2 Enumeration Type Documentation

#### 4.8.2.1 anonymous enum

anonymous enum

#### Enumerator

|        |                  |
|--------|------------------|
| UO_TXD | Possible assign. |
|--------|------------------|

Definition at line 38 of file SwitchMatrix\_FW.h.

```
38     {  
39         UO_TXD,  
40         UO_SCLK,  
41         U1_CTS,  
42         U2_RTS,  
43         SPI0_MOSI,  
44         SPI0_SSEL2,  
45         SPI1_MISO,  
46         SCT_IN1,  
47         SCT_OUT1,  
48         SCT_OUT5,  
49         I2C2_SDA,  
50         COMP0_OUT,  
51         UART3_RXD,  
52         UART4_SCLK,  
53         T0_MAT3  
54     };
```

#### 4.8.2.2 anonymous enum

anonymous enum

#### Enumerator

|        |                  |
|--------|------------------|
| U0_RXD | Possible assign. |
|--------|------------------|

Definition at line 56 of file SwitchMatrix\_FW.h.

```
56     {  
57         U0_RXD,  
58         U1_TXD,  
59         U0_SCLK,  
60         U2_CTS,  
61         SPI0_MISO,  
62         SPI0_SSEL3,  
63         SPI1_SSEL0,  
64         SCT_IN2,  
65         SCT_OUT2,  
66         SCT_OUT6,  
67         I2C2_SCL,  
68         CLKOUT,  
69         UART3_SCLK,  
70         T0_MAT0,  
71         T0_CAP0  
72     };
```

#### 4.8.2.3 anonymous enum

anonymous enum

##### Enumerator

| UO_RTS | Possible assign. |
|--------|------------------|
|--------|------------------|

Definition at line 74 of file SwitchMatrix\_FW.h.

```
74     {  
75         UO_RTS,  
76         U1_RXD,  
77         U2_TXD,  
78         U2_SCLK,  
79         SPI0_SSEL0,  
80         SPI1_SCK,  
81         SPI1_SSEL1,  
82         SCT_IN3,  
83         SCT_OUT3,  
84         I2C1_SDA,  
85         I2C3_SDA,  
86         GPIO_INT_BMAT,  
87         UART4_TXD,  
88         T0_MAT1,  
89         T0_CAP1  
90     };
```

#### 4.8.2.4 anonymous enum

anonymous enum

##### Enumerator

| UO_CTS | Possible assign. |
|--------|------------------|
|--------|------------------|

Definition at line 92 of file SwitchMatrix\_FW.h.

```
92     {  
93         UO_CTS,  
94         U1_RTS,  
95         UO_RXD,  
96         SPI0_SCK,  
97         SPI0_SSEL1,
```

```

98     SPI1_MOSI,
99     SCT0_IN0,
100    SCT_OUT0,
101    SCT_OUT4,
102    I2C1_SCL,
103    I2C3_SCL,
104    UART3_TXD,
105    UART4_RXD,
106    T0_MAT2,
107    T0_CAP2
108 };

```

## 4.8.3 Function Documentation

### 4.8.3.1 SWM()

```

void SWM (
    uint8_t port,
    uint8_t pin,
    uint8_t assign,
    uint8_t byte )

```

: Assign movable functions for pin

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|                       |                         |
|-----------------------|-------------------------|
| [in] uint8_t port :   | PORT0,PORT1             |
| [in] uint8_t pin :    | 0,31                    |
| [in] uint8_t assign : |                         |
| [in] uint8_t byte :   | BYTE0,BYTE1,BYTE2,BYTE3 |

#### Returns

: void

Definition at line 22 of file SwitchMatrix\_FW.c.

```

22
23     pin = pin + 0x20 * port; //PIO0[0:31] 0x00 to 0x1F PIO1[0:21] 0x1F to 0x35
24     PINASSIGN[assign] |= (pin « byte);
25 }

```

#### 4.8.3.2 SWM\_Disable()

```
void SWM_Disable (
    void )
```

: Disable SWM

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: void

Definition at line 67 of file SwitchMatrix\_FW.c.

```
67         {
68     SYSAHBCLKCTRL0&= (~ (1<<7));
69 }
```

#### 4.8.3.3 SWM\_Enable()

```
void SWM_Enable (
    void )
```

: Enable SWM

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 54 of file SwitchMatrix\_FW.c.

```

54      {
55          SYSAHBCLKCTRL0|= (1<<7);
56      }

```

**4.8.3.4 SWM\_PinEnable()**

```

void SWM_PinEnable (
    uint8_t port,
    uint8_t pin,
    uint8_t ena )

```

: Enable pin works as value passed in ena

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                   |                                                        |
|-------------------|--------------------------------------------------------|
| [in] uint8_t port | : PORT0,PORT1                                          |
| [in] uint8_t pin  | : 0,31                                                 |
| [in] uint8_t ena  | : READ Page 143 UserManual. There are multiple choices |

**Returns**

: void

Definition at line 38 of file SwitchMatrix\_FW.c.

```

38      {
39          if (port == PORT1)          //PIENABLE[0] -> PIO0_0 .... PIO1_2
40              if (pin < 3)            //PIENABLE10] -> PIO1_3 .... PIO1_21
41                  port = PORT0;
42          PINENABLE[port] &= ~(1 << ena);
43      }

```

**4.9 inc/SYSCON\_FW.h File Reference**

: Firmware functions for SYSCON

## Macros

- #define **SYSCON\_ADD** ( ( \_\_RW uint32\_t \*) 0x40048000UL)
- #define **SYSMEMREMAP** SYSCON\_ADD [0]
- #define **SYSPLLCTRL** SYSCON\_ADD [2]
- #define **SYSPLLSTAT** SYSCON\_ADD [3]
- #define **SYSOSCCTRL** SYSCON\_ADD [8]
- #define **WDTOSCCTRL** SYSCON\_ADD [9]
- #define **FROOSCCTRL** SYSCON\_ADD [10]
- #define **FRODIRECTCLKUEN** SYSCON\_ADD [12]
- #define **SYSRSTSTAT** SYSCON\_ADD [14]
- #define **SYSPLLCLKSEL** SYSCON\_ADD [16]
- #define **SYSPLLCLKUEN** SYSCON\_ADD [17]
- #define **MAINCLKPLLSEL** SYSCON\_ADD [18]
- #define **MAINCLKPLLUEN** SYSCON\_ADD [19]
- #define **MAINCLKSEL** SYSCON\_ADD [20]
- #define **MAINCLKUEN** SYSCON\_ADD [21]
- #define **SYSAHBCLKDIV** SYSCON\_ADD [22]
- #define **CAPTCLKSEL** SYSCON\_ADD [24]
- #define **ADCCLKSEL** SYSCON\_ADD [25]
- #define **ADCCLKDIV** SYSCON\_ADD [26]
- #define **SCTCLKSEL** SYSCON\_ADD [27]
- #define **SCTCLKDIV** SYSCON\_ADD [28]
- #define **EXTCLKSEL** SYSCON\_ADD [29]
- #define **\_SYSAHBCLKCTRL0** SYSCON\_ADD [32]
- #define **\_SYSAHBCLKCTRL1** SYSCON\_ADD [33]
- #define **PRESETCTRL0** SYSCON\_ADD [34]
- #define **PRESETCTRL1** SYSCON\_ADD [35]
- #define **UART0CLKSEL** SYSCON\_ADD [36]
- #define **UART1CLKSEL** SYSCON\_ADD [37]
- #define **UART2CLKSEL** SYSCON\_ADD [38]
- #define **UART3CLKSEL** SYSCON\_ADD [39]
- #define **UART4CLKSEL** SYSCON\_ADD [40]
- #define **I2C0CLKSEL** SYSCON\_ADD [41]
- #define **I2C1CLKSEL** SYSCON\_ADD [42]
- #define **I2C2CLKSEL** SYSCON\_ADD [43]
- #define **I2C3CLKSEL** SYSCON\_ADD [44]
- #define **SPI0CLKSEL** SYSCON\_ADD [45]
- #define **SPI1CLKSEL** SYSCON\_ADD [46]
- #define **FRG0DIV** SYSCON\_ADD [52]
- #define **FRG0MULT** SYSCON\_ADD [53]
- #define **FRG0CLKSEL** SYSCON\_ADD [54]
- #define **FRG1DIV** SYSCON\_ADD [56]
- #define **FRG1MULT** SYSCON\_ADD [57]
- #define **FRG1CLKSEL** SYSCON\_ADD [58]
- #define **CLKOUTSEL** SYSCON\_ADD [60]
- #define **CLKOUTDIV** SYSCON\_ADD [61]
- #define **EXTTRACECMD** SYSCON\_ADD [63]
- #define **PIOPORCAP0** SYSCON\_ADD [64]
- #define **PIOPORCAP1** SYSCON\_ADD [65]
- #define **\_IOCONCLKDIV6** SYSCON\_ADD [77]
- #define **\_IOCONCLKDIV5** SYSCON\_ADD [78]
- #define **\_IOCONCLKDIV4** SYSCON\_ADD [79]
- #define **\_IOCONCLKDIV3** SYSCON\_ADD [80]
- #define **\_IOCONCLKDIV2** SYSCON\_ADD [81]

- #define **\_IOCONCLKDIV1** SYSCON\_ADD [82]
- #define **\_IOCONCLKDIV0** SYSCON\_ADD [83]
- #define **BODCTRL** SYSCON\_ADD [84]
- #define **SYSTCKCAL** SYSCON\_ADD [85]
- #define **IRQLATENCY** SYSCON\_ADD [92]
- #define **NMISRC** SYSCON\_ADD [93]
- #define **PINTSEL0** SYSCON\_ADD [94]
- #define **PINTSEL1** SYSCON\_ADD [95]
- #define **PINTSEL2** SYSCON\_ADD [96]
- #define **PINTSEL3** SYSCON\_ADD [97]
- #define **PINTSEL4** SYSCON\_ADD [98]
- #define **PINTSEL5** SYSCON\_ADD [99]
- #define **PINTSEL6** SYSCON\_ADD [100]
- #define **PINTSEL7** SYSCON\_ADD [101]
- #define **STARTERP0** SYSCON\_ADD [129]
- #define **STARTERP1** SYSCON\_ADD [133]
- #define **PDSLEEPCFG** SYSCON\_ADD [140]
- #define **PDWAKECFG** SYSCON\_ADD [141]
- #define **PDRUNCFG** SYSCON\_ADD [142]
- #define **DEVICE\_ID** SYSCON\_ADD [254]
- #define **CLOCK\_FRO\_SETTING\_API\_ROM\_ADDRESS** 0x0F0026F5U
- #define **F30MHz** 30000U
- #define **FRO\_OUT\_PowerDown** 1
- #define **FRO\_PD** 2
- #define **SYSCON\_FROOSCCTRL\_FRO\_DIRECT\_MASK** (0x20000U)
- #define **SYSCON\_FROOSCCTRL\_FRO\_DIRECT\_SHIFT** (17U)
- #define **kCLOCK\_FroSrcFroOsc** 1U << SYSCON\_FROOSCCTRL\_FRO\_DIRECT\_SHIFT
- #define **kPDRUNCFG\_PD\_SYSOSC** 0x20
- #define **CLK\_FROM\_SYS\_OSC** 0x00
- #define **FREQ30MHz** 30000000U
- #define **CLK\_SYS\_PLLSRCFRODIV** 0x03
- #define **CLOCK\_FAIM\_BASE** 0x50010000U
- #define **SYSPLL\_MIN\_FCCO\_FREQ\_HZ** 156000000U
- #define **SYSCON\_SYSPLLCTRL\_MSEL\_MASK** 0x1FU
- #define **SYSCON\_SYSPLLCTRL\_MSEL\_SHIFT** (0U)
- #define **SYSCON\_SYSPLLCTRL\_PSEL\_MASK** 0x60U
- #define **SYSCON\_SYSPLLCTRL\_PSEL\_SHIFT** (5U)
- #define **SYSCON\_SYSPLLCTRL\_MSEL(x)** (((uint32\_t)((uint32\_t)(x)) << SYSCON\_SYSPLLCTRL\_MSEL\_SHIFT)) & SYSCON\_SYSPLLCTRL\_MSEL\_MASK
- #define **SYSCON\_SYSPLLCTRL\_PSEL(x)** (((uint32\_t)((uint32\_t)(x)) << SYSCON\_SYSPLLCTRL\_PSEL\_SHIFT)) & SYSCON\_SYSPLLCTRL\_PSEL\_MASK
- #define **CLK\_MAIN\_CLK\_MUX\_GET\_MUX(x)** ((uint32\_t)(x) & 0xFFU)
- #define **CLK\_MAIN\_CLK\_MUX\_GET\_PRE\_MUX(x)** (((uint32\_t)(x) >> 8U) & 0xFFU)
- #define **SYSCON\_MAINCLKSEL\_SEL\_MASK** 0x03U
- #define **SYSCON\_MAINCLKSEL\_SEL\_SHIFT** (0U)
- #define **SYSCON\_MAINCLKSEL\_SEL(x)** (((uint32\_t)((uint32\_t)(x)) << SYSCON\_MAINCLKSEL\_SEL\_SHIFT)) & SYSCON\_MAINCLKSEL\_SEL\_MASK
- #define **SYSCON\_MAINCLKPLLSEL\_SEL\_MASK** (0x3U)
- #define **SYSCON\_MAINCLKPLLSEL\_SEL\_SHIFT** (0U)
- #define **SYSCON\_MAINCLKPLLSEL\_SEL(x)** (((uint32\_t)((uint32\_t)(x)) << SYSCON\_MAINCLKPLLSEL\_SEL\_SHIFT)) & SYSCON\_MAINCLKPLLSEL\_SEL\_MASK
- #define **kCLOCK\_MainClkSrcFro** 0
- #define **SYSCON\_SYSAHBCLKDIV\_DIV(x)** (((uint32\_t)((uint32\_t)(x)) << SYSCON\_SYSAHBCLKDIV\_DIV\_SHIFT)) & SYSCON\_SYSAHBCLKDIV\_DIV\_MASK
- #define **SYSCON\_SYSAHBCLKDIV\_DIV\_MASK** 0xFFU
- #define **SYSCON\_SYSAHBCLKDIV\_DIV\_SHIFT** (0U)



## Functions

- void **BoardClockRUN** ()  
: *Runs clock at 30MHz*
- void **ClockSetFroOscFREQ** (uint32\_t freq)
- void **PowerDisablePD** (uint8\_t en)
- void **CLOCK\_SetFroOutClkSrc** (uint32\_t src)
- void **CLOCK\_Select** (uint8\_t sel)
- void **CLOCK\_InitSystemPll** (uint32\_t freq, uint8\_t src)
- uint32\_t **CLOCK\_GetSystemPLLInClockRate** (void)
- uint32\_t **CLOCK\_GetFroFreq** (void)
- uint32\_t **FindSystemPIIPsel** (uint32\_t outFreq)
- void **CLOCK\_SetMainClkSrc** (uint32\_t src)
- void **CLOCK\_SetCoreSysClkDiv** (uint32\_t value)

### 4.9.1 Detailed Description

: Firmware functions for SYSCON

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

### 4.9.2 Function Documentation

#### 4.9.2.1 BoardClockRUN()

```
void BoardClockRUN (
    void )
```

: Runs clock at 30MHz

: Select clock from fro

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 19 of file SYSCON\_FW.c.

```

19      {
20      PowerDisablePD (FRO_OUT_PowerDown);
21      PowerDisablePD (FRO_PD);
22      ClockSetFroOscFREQ (F30MHz);
23      CLOCK_SetFroOutClkSrc (kCLOCK_FroSrcFroOsc);
24      PowerDisablePD (kPDRUNCFG_PD_SYSSOSC);
25      CLOCK_Select (CLK_FROM_SYS_OSC);
26      CLOCK_InitSystemPll (FREQ30MHz, CLK_SYS_PLLSRCFRODIV);
27      CLOCK_SetMainClkSrc (kCLOCK_MainClkSrcFro);
28      CLOCK_SetCoreSysClkDiv (1U);
29  }
```

**4.10 inc/SysTick\_FW.h File Reference**

: Firmware functions for SysTick

**Macros**

- `#define TICK_OUT_1S 100`  
*Systick interrupt each 1 second.*
- `#define SysTick_ ( ( __RW uint32_t *) 0xE000E000UL)`
- `#define SYST_CSR SysTick_[4]`
- `#define SYST_RVR SysTick_[5]`
- `#define SYST_CVR SysTick_[6]`
- `#define SYST_CALIB SysTick_[7]`
- `#define SYSTICK_ENABLE_INTERRUPT_CLK 0x07`
- `#define SYSTICK_DISABLE 0x00`
- `#define SYSTICK_INT_DIS SYST_CSR &= ~0x02;`
- `#define SYSTICK_INT_EN SYST_CSR = SYSTICK_ENABLE_INTERRUPT_CLK;`
- `#define FRE30MHz 30000U`

**Functions**

- void [SysTick\\_Init](#) (void)  
: Initialize the systick
- void [SysTick\\_Off](#) (void)  
: Stops the systick
- void [SysTick\\_Set](#) (uint32\_t freq)  
: Set the counter as freq\*10mS -1

### 4.10.1 Detailed Description

: Firmware functions for SysTick

: Used for 30 MHz

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

### 4.10.2 Function Documentation

#### 4.10.2.1 SysTick\_Init()

```
void SysTick_Init (  
    void )
```

: Initialize the systick

: Enable SysTick, enable interrupt and set the counter

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

#### Returns

: void

Definition at line 19 of file SysTick\_FW.c.

```
19     {  
20         SysTick_Set(FRE30MHz);  
21         SYST_CSR = SYSTICK_ENABLE_INTERRUPT_CLK;  
22         SYST_CVR = 0;  
23     }
```

#### 4.10.2.2 SysTick\_Off()

```
void SysTick_Off (
    void )
```

: Stops the systick

: disable SysTick, disable interrupt

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: void

Definition at line 34 of file SysTick\_FW.c.

```
34         {
35     SYST_CSR = SYSTICK_DISABLE;
36 }
```

#### 4.10.2.3 SysTick\_Set()

```
void SysTick_Set (
    uint32_t freq )
```

: Set the counter as freq\*10mS -1

: Always use at 30MHz

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

#### Parameters

|  |                              |
|--|------------------------------|
|  | [in] uint32_t freq: FRE30MHz |
|--|------------------------------|

#### Returns

: void

Definition at line 47 of file SysTick\_FW.c.

```
47 {  
48     SYST_RVR = freq*10 - 1; // 30MHz*10mS-1  
49 }
```

## 4.11 source/05-ConversionADC.c File Reference

: Firmware functions ADC

```
#include "Aplication.h"
```

### Functions

- int [main](#) (void)  
: *Main Function*

### Variables

- uint32\_t [tick](#) = 0  
: *Var for SysTick\_Handler.*
- uint32\_t [conv](#) = 0  
: *Var for ADC.*

#### 4.11.1 Detailed Description

: Firmware functions ADC

: 12 bits conversion

#### Author

: Tobias Bavasso Piizzi

#### Date

: 08/01/2021

## 4.11.2 Function Documentation

### 4.11.2.1 main()

```
:int main (
        void )
```

: Main Function

: initialize the system and stay in the while

#### Author

: Tobias Bavasso Piizzi

#### Date

: 08/01/2021

#### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

#### Returns

: int

Definition at line 23 of file 05-ConversionADC.c.

```
23      {
24          uint8_t state = 1;
25          LPC_Init();
26          ADC_Init(ADC_0);
27
28          while(1) {
29
30              if(state == 0){
31                  if ( conv > (0xFFF/2)){
32                      state ++;
33                      GPIO_SetPIN(LedGREEN, LED_OFF);
34                      GPIO_SetPIN(LedBLUE, LED_ON);
35                  }
36              }
37              else if (state == 1){
38                  if ( conv <= (0xFFF/2)){
39                      state --;
40                      GPIO_SetPIN(LedGREEN, LED_ON);
41                      GPIO_SetPIN(LedBLUE, LED_OFF);
42                  }
43              }
44          }
45          return 0 ;
46      }
47 }
```

## 4.11.3 Variable Documentation

#### 4.11.3.1 tick

```
uint32_t tick = 0
```

Var for SysTick\_Handler.

Declared in main.

Definition at line 20 of file 05-ConversionADC.c.

## 4.12 source/ADC\_FW.c File Reference

: Firmware functions ADC

```
#include "Aplication.h"
```

### Functions

- void [ADC\\_Init](#) (uint8\_t port, uint8\_t pin, uint8\_t ena)  
: Initialize ADC on a pin
- void [ADC\\_Power](#) (void)  
: Power ADC
- void [ADC\\_Enable](#) (void)  
: Enable clock in ADC
- void [ADC\\_Disable](#) (void)  
: Disable clock in ADC
- void [ADC0\\_SEQA\\_IRQHandler](#) (void)  
: Interruption for ADC

### Variables

- uint32\_t [tick](#)  
Var for SysTick\_Handler.
- uint32\_t [conv](#)  
Var for ADC.

#### 4.12.1 Detailed Description

: Firmware functions ADC

: 12 bits conversion

Author

: Tobias Bavasso Piizzi

Date

: 08/01/2021

## 4.12.2 Function Documentation

### 4.12.2.1 ADC0\_SEQA\_IRQHandler()

```
:void ADC0_SEQA_IRQHandler (
    void )
```

: Interruption for ADC

: Interrupt when some channel finishes its conversion

#### Author

: Tobias Bavasso Piizzi

#### Date

: 10/01/2021

#### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

#### Returns

: void

< Clean flags

< Read global data

< Make an average

< Start a new conversion

Definition at line 104 of file ADC\_FW.c.

```
104     {
105         static uint8_t i = 0;
106         static uint32_t sum = 0;
107
108         (void) _ADC_SEQA_GDAT;
109
110         sum += _ADC_SEQA_GDAT->_RESULT;
111         i++;
112         if( i == 0xFF ){
113             conv = sum/i ;
114             i = 0;
115             sum = 0;
116         }
117         _ADC_SEQA_CTRL->_START = 1;
118     }
```



#### 4.12.2.2 ADC\_Disable()

```
:void ADC_Disable (  
    void )
```

: Disable clock in ADC

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 08/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: void

Definition at line 90 of file ADC\_FW.c.

```
90      {  
91      SYSAHBCLKCTRL0&= (~ (1<<ADC_SYSAHB)) ;  
92  }
```

#### 4.12.2.3 ADC\_Enable()

```
:void ADC_Enable (  
    void )
```

: Enable clock in ADC

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 08/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 77 of file ADC\_FW.c.

```
77      {  
78      SYSAHBCLKCTRL0|= (1<<ADC_SYSAHB);  
79  }
```

**4.12.2.4 ADC\_Init()**

```
:void ADC_Init (  
    uint8_t port,  
    uint8_t pin,  
    uint8_t ena )
```

: Initialize ADC on a pin

: Continuous conversion of POTE in board

**Author**

: Tobias Bavasso Piizzi

**Date**

: 08/01/2021

**Parameters**

|  |                                                           |
|--|-----------------------------------------------------------|
|  | [in] uint8_t port: PORT0,PORT1                            |
|  | [in] uint8_t pin: 0,31                                    |
|  | [in] uint8_t en: bit to enable in PINENABLE (page 143 UM) |

**Returns**

: void

< Enable CLOCK in SYSAHB

< Enable service interrupt

< Interrupt after conversion finish

< Enable Switch Matrix

< Enable pin in SWN as AnalogInput

< Disable Switch Matrix

< Power in SYSCON

< Div = 0

< Sync

< OFF

< OFF

< Sample CH0

< No hardware trigger

< Positive trigger

< Enable sync

< Individual end of conversion

< Start,enable set on the same line first time

Definition at line 23 of file ADC\_FW.c.

```

23                                     {
24
25     ADC_Enable();
26     ISER0|= MASK_ISE_ADC_SEQA;
27     ADC_INTEN|= MASK_SEQA_INTEN;
28     SWM_Enable();
29     SWM_PinEnable(port, pin, ena);
30     SWM_Disable();
31     ADC_Power();
32
33
34
35     ADC_CTRL->_CLKDIV = 0x00;
36     ADC_CTRL->_ASYNCMODE = 0;
37     ADC_CTRL->_LPWRMODE = 0;
38     ADC_CTRL->_CALMODE = 0;
39
40     ADC_SEQA_CTRL->_CHANNELS      = 0x01;
41     ADC_SEQA_CTRL->_TRIGGER       = 0x00;
42     ADC_SEQA_CTRL->_TRIGPOL      = 0x1;
43     ADC_SEQA_CTRL->_SYNCBYPASS    = 0x0;
44     ADC_SEQA_CTRL->_TSAMP        = 0x00;
45     ADC_SEQA_CTRL->_START        = 0;
46     ADC_SEQA_CTRL->_BURST        = 0;
47     ADC_SEQA_CTRL->_SINGLESTEP    = 0x0;
48     ADC_SEQA_CTRL->_LOWPRIO      = 0x0;
49     ADC_SEQA_CTRL->_MODE         = 0;
50     ADC_SEQA_CTRL->_SEQx_ENA     = 0;
51     _ADC_SEQA_CTRL |= ((0b100001) << 26);
52 }
```

#### 4.12.2.5 ADC\_Power()

```

: void ADC_Power (
    void )
```

: Power ADC

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 08/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 63 of file ADC\_FW.c.

```

63      {
64      PDRUNCFG&= (~ (1 << MASK_ADC_SYSCON));
65
66 }
```

## 4.13 source/Application.c File Reference

: Functions used in main

```
#include "Aplication.h"
```

**Functions**

- void [LPC\\_Init](#) (void)  
: Initialize the board
- void [GPIO\\_Init](#) (void)  
: Initialize the GPIO

**Variables**

- uint32\_t [tick](#)  
Declared in main.

### 4.13.1 Detailed Description

: Functions used in main

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

## 4.13.2 Function Documentation

### 4.13.2.1 GPIO\_Init()

```
:void GPIO_Init (
    void )
```

: Initialize the GPIO

: It depends on each project

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|           |
|-----------|
| [in] void |
|-----------|

#### Returns

: void

Definition at line 35 of file Application.c.

```
35     {
36         GPIO_SetDIR(UserKEY, INPUT);
37         GPIO_SetDIR(LedGREEN, OUTPUT);
38         GPIO_SetDIR(LedBLUE, OUTPUT);
39
40         GPIO_SetPIN(LedGREEN, LED_OFF);
41         GPIO_SetPIN(LedBLUE, LED_OFF);
42     }
```

### 4.13.2.2 LPC\_Init()

```
:void LPC_Init (
    void )
```

: Initialize the board

: It depends on each project

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 19 of file Aplicacion.c.

```

19      {
20          GPIO_Enable();
21          BoardClockRUN();
22          SysTick_Init();
23          GPIO_Init();
24      }
```

**4.13.3 Variable Documentation****4.13.3.1 tick**

```
uint32_t tick [extern]
```

Declared in main.

Declared in main.

Definition at line 20 of file 05-ConversionADC.c.

**4.14 source/Disp7Seg\_FW.c File Reference**

: Firmware functions for DISP7SEG

```
#include "Aplication.h"
```

**Functions**

- void [DISP7SEG\\_Init](#) (void)  
: Set pins for display as out
- void [DISP\\_Sweep](#) (void)  
: Refresh the display 7Seg (2 Disp)

**Variables**

- `__RW uint8_t` [buff\\_Disp7](#) []  
Display buffer.

### 4.14.1 Detailed Description

: Firmware functions for DISP7SEG

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 07/01/2021

### 4.14.2 Function Documentation

#### 4.14.2.1 DISP7SEG\_Init()

```
:void DISP7SEG_Init (  
    void )
```

: Set pins for display as out

:

#### Author

: Tobias Bavasso Piizzi

#### Date

: 07/01/2021

#### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

#### Returns

: void

Definition at line 19 of file Disp7Seg\_FW.c.

```
19      {  
20      GPIO_SetDIR(SEG_A, OUTPUT);  
21      GPIO_SetDIR(SEG_B, OUTPUT);  
22      GPIO_SetDIR(SEG_C, OUTPUT);  
23      GPIO_SetDIR(SEG_D, OUTPUT);  
24      GPIO_SetDIR(SEG_E, OUTPUT);  
25      GPIO_SetDIR(SEG_F, OUTPUT);
```

```

26     GPIO_SetDIR(SEG_G, OUTPUT);
27     GPIO_SetDIR(TR_D0, OUTPUT);
28     GPIO_SetDIR(TR_D1, OUTPUT);
29
30     GPIO_ClearOUT(SEG_A);
31     GPIO_ClearOUT(SEG_B);
32     GPIO_ClearOUT(SEG_C);
33     GPIO_ClearOUT(SEG_D);
34     GPIO_ClearOUT(SEG_E);
35     GPIO_ClearOUT(SEG_F);
36     GPIO_ClearOUT(SEG_G);
37     GPIO_ClearOUT(TR_D0);
38     GPIO_ClearOUT(TR_D1);
39 }

```

#### 4.14.2.2 DISP\_Sweep()

```

: void DISP_Sweep (
    void )

```

: Refresh the display 7Seg (2 Disp)

: Is necessary to be used in SysTick\_Handler

##### Author

: Tobias Bavasso Piizzi

##### Date

: 07/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: void

< Number of disp

< Turn off transistor

< Turn off transistor

< Next time sweep other disp

< Reset the digits

Definition at line 51 of file Disp7Seg\_FW.c.

```

51     {
52     uint8_t aux;
53     static uint8_t digit = 0;
54
55     GPIO_ClearOUT(TR_D0);
56     GPIO_ClearOUT(TR_D1);

```



```

57
58     aux = buff_Dis7[digit];
59
60     GPIO_SetPIN( SEG_A, ((aux >> 0) & (uint8_t) 0x01));
61     GPIO_SetPIN( SEG_B, ((aux >> 1) & (uint8_t) 0x01));
62     GPIO_SetPIN( SEG_C, ((aux >> 2) & (uint8_t) 0x01));
63     GPIO_SetPIN( SEG_D, ((aux >> 3) & (uint8_t) 0x01));
64     GPIO_SetPIN( SEG_E, ((aux >> 4) & (uint8_t) 0x01));
65     GPIO_SetPIN( SEG_F, ((aux >> 5) & (uint8_t) 0x01));
66     GPIO_SetPIN( SEG_G, ((aux >> 6) & (uint8_t) 0x01));
67     GPIO_SetPIN( SEG_DP, ((aux >> 7) & (uint8_t) 0x01));
68
69     switch (digit) {
70     case DIGIT_0:
71         GPIO_SetOUT(TR_D0);
72         break;
73     case DIGIT_1:
74         GPIO_SetOUT(TR_D1);
75         break;
76     default:
77         digit = 0;
78         GPIO_SetOUT(TR_D0);
79         break;
80     }
81
82     digit++;
83     digit %= DIGITS;
84
85 }

```

## 4.15 source/Disp7Seg\_SW.c File Reference

: Software functions for DISP7SEG

```
#include "Aplication.h"
```

### Functions

- void [Display](#) (uint8\_t val)  
: Writes on Disp7Seg

### Variables

- \_\_RW uint8\_t [buff\\_Dis7](#) [DIGITS]  
Buffer de display.
- uint8\_t [Digits\\_to\\_BCD7seg](#) []
- \_\_RW uint8\_t [tick\\_Dis7](#)

### 4.15.1 Detailed Description

: Software functions for DISP7SEG

:

Author

: Tobias Bavasso Piizzi

Date

: 07/01/2021

## 4.15.2 Function Documentation

### 4.15.2.1 Display()

```
:void Display (
    uint8_t val )
```

: Writes on Disp7Seg

: High lever of layers

#### Author

: Tobias Bavasso Piizzi

#### Date

: 07/01/2021

#### Parameters

|  |                           |
|--|---------------------------|
|  | [in] uint8_t val: 0 to 99 |
|--|---------------------------|

#### Returns

: void

< Disable SysTick INT

< Enable SysTick INT

Definition at line 38 of file Disp7Seg\_SW.c.

```
38     {
39         uint8_t i;
40         uint8_t auxDisp[DIGITS];
41
42         for (i = 0; i < DIGITS; i++) {
43             auxDisp[i] = Digits_to_BCD7seg[val % 10];
44             val /= 10;
45         }
46         for (i = 0; i < DIGITS; i++) {
47             SYSTICK_INT_DIS;
48             buff_Disp7[i] = auxDisp[i];
49             SYSTICK_INT_EN;
50         }
51     }
52 }
```

## 4.15.3 Variable Documentation

#### 4.15.3.1 Digits\_to\_BCD7seg

```
uint8_t Digits_to_BCD7seg[]
```

##### Initial value:

```
= { 0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7c, 0x07,  
    0x7f, 0x67 }
```

Tabla de conversion bcd a 7 segmentos  
Codigo bcd a b c d e f g dp  
0 1 1 1 1 1 1 0 0 1 0 1 1 0 0 0 0 2 1 1 0 1 1 0 1  
3 1 1 1 1 0 0 1 4 0 1 1 0 0 1 1 5 1 0 1 1 0 1 1 6 0 0 1 1 1 1 1 7 1 1 1 0 0 0 0 8 1 1 1 1 1 1 1 9 1 1 1 0 0 1 1

Definition at line 26 of file Disp7Seg\_SW.c.

## 4.16 source/GPIO\_FW.c File Reference

: Firmware functions for GPIO

```
#include "Aplication.h"
```

### Functions

- void [GPIO\\_Enable](#) (void)  
: Enable GPIO0 and GPIO1
- void [GPIO\\_Disable](#) (void)  
: Disable GPIO0 and GPIO1
- void [GPIO\\_SetDIR](#) (uint8\_t port, uint8\_t pin, uint8\_t dir)  
: Choose GPIO as Input/Output
- void [GPIO\\_SetPIN](#) (uint8\_t port, uint8\_t pin, uint8\_t state)  
: Choose GPIO's output state
- uint8\_t [GPIO\\_GetPIN](#) (uint8\_t port, uint8\_t pin, uint8\_t state)  
: Return GPIO's input state
- void [GPIO\\_SetOUT](#) (uint8\_t port, uint8\_t pin)  
: Put GPIO's out to 1
- void [GPIO\\_ClearOUT](#) (uint8\_t port, uint8\_t pin)  
: Put GPIO's out to 0
- void [GPIO\\_ToggleOUT](#) (uint8\_t port, uint8\_t pin)  
: Invert GPIO's out
- void [GPIO\\_DebounceUserKEY](#) (void)  
: Firmware debounce for user key in board
- void [GPIO\\_Debounce](#) (uint8\_t port, uint8\_t pin, uint8\_t state)  
: Firmware debounce for a GPIO
- void [IOCONEnable](#) (void)  
: Enable IOCON
- void [IOCONDisable](#) (void)  
: Disable IOCON
- uint8\_t [GetOFFSET](#) (uint8\_t port, uint8\_t pin)  
: Usefull for SetMode functions
- void [GPIO\\_SetModeINPUT](#) (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: on-chip pull-up/pull-down resistor

- void `GPIO_SetModeHYS` (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: *Hysteresis*
- void `GPIO_SetModeINV` (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: *Invert input*
- void `GPIO_SetModeOD` (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: *Open drain*
- void `GPIO_SetModeFILTER` (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: *Digital filter sample mode*
- void `GPIO_SetModeCLKDIV` (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: *Select peripheral clock divider for input filter sampling clock*
- void `GPIO_SetModeDAC` (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: *Selects DAC mode*
- void `GPIO_SetModeI2C` (uint8\_t port, uint8\_t pin, uint8\_t mode)  
: *Selects I2C mode*

## Variables

- \_\_RW uint8\_t `buff_UserKEY` = 0
- \_\_RW uint8\_t `buff_In` = 0
- uint8\_t `offset` []

### 4.16.1 Detailed Description

: Firmware functions for GPIO

:

Author

: Tobias Bavasso Piizzi

Date

: 04/01/2021

### 4.16.2 Function Documentation

#### 4.16.2.1 GetOFFSET()

```
:uint8_t GetOFFSET (
    uint8_t port,
    uint8_t pin )
```

: Usefull for SetMode functions

:

Author

: Tobias Bavasso Piizzi

Date

: 04/01/2021

**Parameters**

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |

**Returns**

: void

Definition at line 231 of file GPIO\_FW.c.

```
231 {
232     uint8_t index;
233     index = port * 32 + pin;
234     return ((offset[index]) / 4);
235 }
```

**4.16.2.2 GPIO\_ClearOUT()**

```
:void GPIO_ClearOUT (
    uint8_t port,
    uint8_t pin )
```

: Put GPIO's out to 0

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |

**Returns**

: void

Definition at line 113 of file GPIO\_FW.c.

```
113 {
114     GPIO_CLRP[port] |= (1 « pin);
115 }
```

#### 4.16.2.3 GPIO\_Debounce()

```
:void GPIO_Debounce (
    uint8_t port,
    uint8_t pin,
    uint8_t state )
```

: Firmware debounce for a GPIO

: Use in SysTick\_Handler or in some timer interrupt

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|                      |                  |
|----------------------|------------------|
| [in] uint8_t port :  | PORT0,PORT1      |
| [in] uint8_t pin :   | 0,31             |
| [in] uint8_t state : | ACT_LOW,ACT_HIGH |

##### Returns

: void

Definition at line 169 of file GPIO\_FW.c.

```
169
170     static uint8_t q = 0;    //Quantity of bounces
171     uint8_t j = 0;          //It captures changes
172
173     if (GPIO_GetPIN(port, pin, state))    // The key is pushed?
174         j = 0x01;                        //Something is happening, the key is been pushed
175
176     if (buff_In ^ j) {                  // If the key is pushed while q != BOUNCE
177         q++;                            // I change the buffer
178         if (q == BOUNCE) {
179             q = 0;
180             buff_In ^= 0x01;
181         }
182     } else
183         q = 0;
184 }
```

#### 4.16.2.4 GPIO\_DebounceUserKEY()

```
:void GPIO_DebounceUserKEY (
    void )
```

: Firmware debounce for user key in board

: Use in SysTick\_Handler or in some timer interrupt

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |      |
|--|------|
|  | [in] |
|--|------|

**Returns**

: void

Definition at line 141 of file GPIO\_FW.c.

```

141     {
142         static uint8_t q = 0;    //Quantity of bounces
143         uint8_t j = 0;          //It captures changes
144
145         if (GPIO_GetPIN(UserKEY, ACT_LOW))    // The key is pushed?
146             j = 0x01;                        //Something is happening, the key is been pushed
147
148         if (buff_UserKEY ^ j) {                // If the key is pushed while q != BOUNCE
149             q++;                               // I change the buffer
150             if (q == BOUNCE) {
151                 q = 0;
152                 buff_UserKEY ^= 0x01;
153             }
154         } else
155             q = 0;
156     }

```

**4.16.2.5 GPIO\_Disable()**

```

: void GPIO_Disable (
    void )

```

: Disable GPIO0 and GPIO1

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 32 of file GPIO\_FW.c.

```

32     {
33         SYSAHBCLKCTRL0 &= (~ (1<<6));
34         SYSAHBCLKCTRL0 &= (~ (1<<20));
35     }
```

**4.16.2.6 GPIO\_Enable()**

```

: void GPIO_Enable (
    void )
```

: Enable GPIO0 and GPIO1

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|           |
|-----------|
| [in] void |
|-----------|

**Returns**

: void

Definition at line 19 of file GPIO\_FW.c.

```

19     {
20         SYSAHBCLKCTRL0 |= (1<<6);
21         SYSAHBCLKCTRL0 |= (1<<20);
22     }
```

**4.16.2.7 GPIO\_GetPIN()**

```

: uint8_t GPIO_GetPIN (
    uint8_t port,
    uint8_t pin,
    uint8_t dir )
```

: Return GPIO's input state

:



**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                    |                    |
|--------------------|--------------------|
| [in] uint8_t port  | : PORT0,PORT1      |
| [in] uint8_t pin   | : 0,31             |
| [in] uint8_t STATE | : ACT_LOW,ACT_HIGH |

**Returns**

: uint8\_t : 1 pin == [state] , 0 pin != [state]

Definition at line 81 of file GPIO\_FW.c.

```

81                                     {
82     port = port * 32 + pin;
83     if ( GPIO_PBYTE[port] == state)
84         return 1;
85     else
86         return 0;
87 }
```

**4.16.2.8 GPIO\_SetDIR()**

```

: void GPIO_SetDIR (
    uint8_t port,
    uint8_t pin,
    uint8_t dir )
```

: Choose GPIO as Input/Output

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                   |                |
|-------------------|----------------|
| [in] uint8_t port | : PORT0,PORT1  |
| [in] uint8_t pin  | : 0,31         |
| [in] uint8_t dir  | : INPUT,OUTPUT |

**Returns**

: void

Definition at line 48 of file GPIO\_FW.c.

```

48                                     {
49     GPIO_DIRP[port] &= (~(1 « pin));
50     GPIO_DIRP[port] |= (dir « pin);
51 }
```

**4.16.2.9 GPIO\_SetModeCLKDIV()**

```

: void GPIO_SetModeCLKDIV (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Select peripheral clock divider for input filter sampling clock

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                                                                                                           |
|-----------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: IOCONCLKDIV0 to IOCONCLKDIV6 |
|-----------------------------------------------------------------------------------------------------------|

**Returns**

: void

Definition at line 338 of file GPIO\_FW.c.

```

338                                     {
339     uint8_t offset;
340     offset = GetOFFSET(port, pin);
341     IOCON_[offset] &= (~(0x07 « 13));
342     IOCON_[offset] |= (mode « 13);
343 }
```

**4.16.2.10 GPIO\_SetModeDAC()**

```

: void GPIO_SetModeDAC (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Selects DAC mode

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                                                                                             |
|---------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: DAC_EN,DAC_DIS |
|---------------------------------------------------------------------------------------------|

**Returns**

: void

Definition at line 356 of file GPIO\_FW.c.

```

356                                     {
357     uint8_t offset;
358     offset = GetOFFSET(port, pin);
359     IOCON_[offset] &= (~(0x01 « 16));
360     IOCON_[offset] |= (mode « 16);
361 }
```

**4.16.2.11 GPIO\_SetModeFILTER()**

```

: void GPIO_SetModeFILTER (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Digital filter sample mode

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: BYPASS_FILTER,CLK1_FILTER,CLK2_FILTER,CLK3_FILTER |
|--------------------------------------------------------------------------------------------------------------------------------|

**Returns**

: void

Definition at line 320 of file GPIO\_FW.c.

```

320                                     {
321     uint8_t offset;
322     offset = GetOFFSET(port, pin);
323     IOCON[offset] &= (~(0x03 « 11));
324     IOCON[offset] |= (mode « 11);
325 }
```

**4.16.2.12 GPIO\_SetModeHYS()**

```

: void GPIO_SetModeHYS (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Hysteresis

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                                |                          |                                    |
|--------------------------------|--------------------------|------------------------------------|
| [in] uint8_t port: PORT0,PORT1 | : [in] uint8_t pin: 0,31 | : [in] uint8_t mode:HYS_EN,HYS_DIS |
|--------------------------------|--------------------------|------------------------------------|

**Returns**

: void

Definition at line 266 of file GPIO\_FW.c.

```

266                                     {
267     uint8_t offset;
268     offset = GetOFFSET(port, pin);
269     IOCON[offset] &= (~(0x01 « 5));
270     IOCON[offset] |= (mode « 5);
271 }
```

**4.16.2.13 GPIO\_SetModeI2C()**

```

: void GPIO_SetModeI2C (
    uint8_t port,
```

```
uint8_t pin,  
uint8_t mode )
```

: Selects I2C mode

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                                                                                                         |
|---------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode:STD_MODE,STD_GPIO,FAST_MODE |
|---------------------------------------------------------------------------------------------------------|

**Returns**

: void

Definition at line 374 of file GPIO\_FW.c.

```
374                                     {  
375     uint8_t offset;  
376     offset = GetOFFSET(port, pin);  
377     IOCON_[offset] &= (~(0x03 « 8));  
378     IOCON_[offset] |= (mode « 8);  
379 }
```

**4.16.2.14 GPIO\_SetModeINPUT()**

```
:void GPIO_SetModeINPUT (  
    uint8_t port,  
    uint8_t pin,  
    uint8_t mode )
```

: on-chip pull-up/pull-down resistor

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                                                                                                                        |
|------------------------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode:NO_PULL_UP_DOWN,PULL_DOWN,PULL_UP,REPEATER |
|------------------------------------------------------------------------------------------------------------------------|

**Returns**

: void

Definition at line 248 of file GPIO\_FW.c.

```
248                                     {
249     uint8_t offset;
250     offset = GetOFFSET(port, pin);
251     IOCON[offset] &= (~(0x03 « 3));
252     IOCON[offset] |= (mode « 3);
253 }
```

**4.16.2.15 GPIO\_SetModeINV()**

```
:void GPIO_SetModeINV (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Invert input

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|                                                                                                      |
|------------------------------------------------------------------------------------------------------|
| [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: INV_INPUT,NOT_INV_INPUT |
|------------------------------------------------------------------------------------------------------|

**Returns**

: void

Definition at line 284 of file GPIO\_FW.c.

```
284                                     {
285     uint8_t offset;
286     offset = GetOFFSET(port, pin);
287     IOCON[offset] &= (~(0x01 « 6));
288     IOCON[offset] |= (mode « 6);
289 }
```

#### 4.16.2.16 GPIO\_SetModeOD()

```
:void GPIO_SetModeOD (
    uint8_t port,
    uint8_t pin,
    uint8_t mode )
```

: Open drain

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |                                                                                           |
|--|-------------------------------------------------------------------------------------------|
|  | [in] uint8_t port: PORT0,PORT1 : [in] uint8_t pin: 0,31 : [in] uint8_t mode: OD_EN,OD_DIS |
|--|-------------------------------------------------------------------------------------------|

##### Returns

: void

Definition at line 302 of file GPIO\_FW.c.

```
302                                     {
303     uint8_t offset;
304     offset = GetOFFSET(port, pin);
305     IOCON_[offset] &= (~ (0x01 « 10));
306     IOCON_[offset] |= (mode « 10);
307 }
```

#### 4.16.2.17 GPIO\_SetOUT()

```
:void GPIO_SetOUT (
    uint8_t port,
    uint8_t pin )
```

: Put GPIO's out to 1

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

**Parameters**

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |

**Returns**

: void

Definition at line 99 of file GPIO\_FW.c.

```
99                                     {
100     GPIO_SETP[port] |= (1 << pin);
101 }
```

**4.16.2.18 GPIO\_SetPIN()**

```
:void GPIO_SetPIN (
    uint8_t port,
    uint8_t pin,
    uint8_t dir )
```

: Choose GPIO's output state

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |
|  | [in] uint8_t state : LOW,HIGH   |

**Returns**

: void

Definition at line 64 of file GPIO\_FW.c.

```
64                                     {
65     port = port * 32 + pin;
66     GPIO_PBYTE[port] &= (~1);
67     GPIO_PBYTE[port] |= state;
68 }
```



#### 4.16.2.19 GPIO\_ToogleOUT()

```
:void GPIO_ToogleOUT (
    uint8_t port,
    uint8_t pin )
```

: Invert GPIO's out

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |                                 |
|--|---------------------------------|
|  | [in] uint8_t port : PORT0,PORT1 |
|  | [in] uint8_t pin : 0,31         |

##### Returns

: void

Definition at line 127 of file GPIO\_FW.c.

```
127
128     GPIO_NOTP[port] |= (1 « pin);
129 }
```

#### 4.16.2.20 IOCONDisable()

```
:void IOCONDisable (
    void )
```

: Disable IOCON

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

**Parameters**

|  |      |
|--|------|
|  | [in] |
|--|------|

**Returns**

: void

Definition at line 208 of file GPIO\_FW.c.

```

208     {
209         SYSAHBCLKCTRL0&= (~ (1<<18));
210     }
```

**4.16.2.21 IOCONEnable()**

```

: void IOCONEnable (
        void )
```

: Enable IOCON

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |      |
|--|------|
|  | [in] |
|--|------|

**Returns**

: void

Definition at line 195 of file GPIO\_FW.c.

```

195     {
196         SYSAHBCLKCTRL0|= (1<<18);
197     }
```

**4.16.3 Variable Documentation**

#### 4.16.3.1 offset

```
uint8_t offset[]
```

##### Initial value:

```
= { 0x044, 0x02C, 0x018, 0x014, 0x010, 0x00C, 0x040, 0x03C,  
    0x038, 0x034, 0x020, 0x01C, 0x008, 0x004, 0x048, 0x028, 0x024, 0x000,  
    0x078, 0x074, 0x070, 0x06C, 0x068, 0x064, 0x060, 0x05C, 0x058, 0x054,  
    0x050, 0x0C8, 0x0CC, 0x08C, 0x090, 0x094, 0x098, 0x0A4, 0x0A8, 0x0AC,  
    0x0B8, 0x0C4, 0x07C, 0x080, 0x0DC, 0x0D8, 0x084, 0x088, 0x09C, 0x0A0,  
    0x0B0, 0x0B4, 0x0BC, 0x0C0, 0x0D0, 0x0D4 }
```

Definition at line 214 of file GPIO\_FW.c.

## 4.17 source/GPIO\_SW.c File Reference

: Software functions for GPIO

```
#include "Aplication.h"
```

### Functions

- `uint8_t GetUserKEY` (void)  
: State of the user key in board
- `uint8_t GetInput` (void)  
: State of the input

### Variables

- `uint8_t buff_UserKEY`
- `uint8_t buff_In`

#### 4.17.1 Detailed Description

: Software functions for GPIO

: These functions avoid bouncing. Both must be used w/ GPIO\_DebounceUserKEY or GPIO\_Debounce

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

#### 4.17.2 Function Documentation

#### 4.17.2.1 GetInput()

```
:uint8_t GetInput (
    void )
```

: State of the input

: Is necessary using GPIO\_Debounce

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: uint8\_t 1 if input pressed, 0 if input pressed

Definition at line 48 of file GPIO\_SW.c.

```
48     {
49         static uint8_t buff_before = 0x00;
50
51         if ( buff_In == 0x01 && buff_before == 0x00 ){
52             buff_before = 0x01;
53             return (1);
54         }
55         else if ( buff_In == 0x01 && buff_before == 0x01 )
56             return (0);
57         else if ( buff_In == 0x00 && buff_before == 0x01 )
58             return (0);
59         else
60             return (0);
61 }
```

#### 4.17.2.2 GetUserKEY()

```
:uint8_t GetUserKEY (
    void )
```

: State of the user key in board

: Is necessary using GPIO\_DebounceUserKEY

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

## Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

## Returns

: uint8\_t 1 if user key pressed, 0 if user key not

Definition at line 21 of file GPIO\_SW.c.

```

21     {
22         static uint8_t buff_before = 0x00;
23
24         if ( buff_UserKEY == 0x01 && buff_before == 0x00 ){
25             buff_before = 0x01;
26             return (1);
27         }
28         else if ( buff_UserKEY == 0x01 && buff_before == 0x01 )
29             return (0);
30         else if ( buff_UserKEY == 0x00 && buff_before == 0x01 ){
31             buff_before = 0x00;
32             return (0);
33         }
34         else
35             return (0);
36     }

```

## 4.18 source/mtb.c File Reference

MTB initialization file.

```
#include <cr_mtb_buffer.h>
```

### Macros

- `#define __MTB_BUFFER_SIZE 128`

### Functions

- `__CR_MTB_BUFFER (__MTB_BUFFER_SIZE)`

#### 4.18.1 Detailed Description

MTB initialization file.

Symbols controlling behavior of this code... `__MTB_DISABLE` If this symbol is defined, then the buffer array for the MTB will not be created.

`__MTB_BUFFER_SIZE` Symbol specifying the sizer of the buffer array for the MTB. This must be a power of 2 in size, and fit into the available RAM. The MTB buffer will also be aligned to its 'size' boundary and be placed at the start of a RAM bank (which should ensure minimal or zero padding due to alignment).

`__MTB_RAM_BANK` Allows MTB Buffer to be placed into specific RAM bank. When this is not defined, the "default" (first if there are several) RAM bank is used.

## 4.19 source/SwitchMatrix\_FW.c File Reference

: Firmware functions for SWM

```
#include "Aplication.h"
```

### Functions

- void [SWM](#) (uint8\_t port, uint8\_t pin, uint8\_t assign, uint8\_t byte)  
: *Assign movable functions for pin*
- void [SWM\\_PinEnable](#) (uint8\_t port, uint8\_t pin, uint8\_t ena)  
: *Enable pin works as value passed in ena*
- void [SWM\\_Enable](#) (void)  
: *Enable SWM*
- void [SWM\\_Disable](#) (void)  
: *Disable SWM*

### 4.19.1 Detailed Description

: Firmware functions for SWM

:

Author

: Tobias Bavasso Piizzi

Date

: 04/01/2021

### 4.19.2 Function Documentation

#### 4.19.2.1 SWM()

```
:void SWM (
    uint8_t port,
    uint8_t pin,
    uint8_t assign,
    uint8_t byte )
```

: Assign movable functions for pin

:

Author

: Tobias Bavasso Piizzi

Date

: 04/01/2021

**Parameters**

|  |                                             |
|--|---------------------------------------------|
|  | [in] uint8_t port : PORT0,PORT1             |
|  | [in] uint8_t pin : 0,31                     |
|  | [in] uint8_t assign :                       |
|  | [in] uint8_t byte : BYTE0,BYTE1,BYTE2,BYTE3 |

**Returns**

: void

Definition at line 22 of file SwitchMatrix\_FW.c.

```
22                                     {
23     pin = pin + 0x20 * port; //PIO0[0:31] 0x00 to 0x1F PIO1[0:21] 0x1F to 0x35
24     PINASSIGN[assign] |= (pin « byte);
25 }
```

**4.19.2.2 SWM\_Disable()**

```
:void SWM_Disable (
    void )
```

: Disable SWM

:

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 67 of file SwitchMatrix\_FW.c.

```
67                                     {
68     SYSAHBCLKCTRL0&= (~ (1<<7));
69 }
```

#### 4.19.2.3 SWM\_Enable()

```
:void SWM_Enable (
    void )
```

: Enable SWM

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021

##### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

##### Returns

: void

Definition at line 54 of file SwitchMatrix\_FW.c.

```
54      {
55      SYSAHBCLKCTRL0|= (1<<7);
56 }
```

#### 4.19.2.4 SWM\_PinEnable()

```
:void SWM_PinEnable (
    uint8_t port,
    uint8_t pin,
    uint8_t ena )
```

: Enable pin works as value passed in ena

:

##### Author

: Tobias Bavasso Piizzi

##### Date

: 04/01/2021



## Parameters

|                                                                         |
|-------------------------------------------------------------------------|
| [in] uint8_t port : PORT0,PORT1                                         |
| [in] uint8_t pin : 0,31                                                 |
| [in] uint8_t ena : READ Page 143 UserManual. There are multiple choices |

## Returns

: void

Definition at line 38 of file SwitchMatrix\_FW.c.

```

38                                     {
39     if (port == PORT1)              //PIENABLE[0] -> PIO0_0 .... PIO1_2
40     if (pin < 3)                    //PIENABLE10] -> PIO1_3 .... PIO1_21
41         port = PORT0;
42     PINENABLE[port] &= (~(1 << ena));
43 }
```

## 4.20 source/SYSCON\_FW.c File Reference

: Firmware functions for SYSCON

#include "Aplication.h"

### Functions

- void [BoardClockRUN](#) (void)  
: Runs clock at 30MHz
- void **ClockSetFroOscFREQ** (uint32\_t freq)
- void **PowerDisablePD** (uint8\_t en)
- void **CLOCK\_SetFroOutClkSrc** (uint32\_t src)
- void **CLOCK\_Select** (uint8\_t sel)
- void **CLOCK\_InitSystemPII** (uint32\_t freq, uint8\_t src)
- uint32\_t **CLOCK\_GetSystemPLLInClockRate** (void)
- uint32\_t **CLOCK\_GetFroFreq** (void)
- uint32\_t **FindSyestemPIIPsel** (uint32\_t outFreq)
- void **CLOCK\_SetMainClkSrc** (uint32\_t src)
- void **CLOCK\_SetCoreSysClkDiv** (uint32\_t value)

#### 4.20.1 Detailed Description

: Firmware functions for SYSCON

: Only starts the board at 30MHz

## Author

: Tobias Bavasso Piizzi

## Date

: 04/01/2021

## 4.20.2 Function Documentation

### 4.20.2.1 BoardClockRUN()

```
:void BoardClockRUN (
    void )
```

: Runs clock at 30MHz

: Select clock from fro

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

#### Returns

: void

Definition at line 19 of file SYSCON\_FW.c.

```
19     {
20     PowerDisablePD(FRO_OUT_PowerDown);
21     PowerDisablePD(FRO_PD);
22     ClockSetFroOscFREQ(F30MHz);
23     CLOCK_SetFroOutClkSrc(kCLOCK_FroSrcFroOsc);
24     PowerDisablePD(kPDRUNCFG_PD_SYSOSC);
25     CLOCK_Select(CLK_FROM_SYS_OSC);
26     CLOCK_InitSystemPll(FREQ30MHz, CLK_SYS_PLLSRCFRODIV);
27     CLOCK_SetMainClkSrc(kCLOCK_MainClkSrcFro);
28     CLOCK_SetCoreSysClkDiv(1U);
29 }
```

## 4.21 source/SysTick\_FW.c File Reference

: Firmware functions for SysTick

```
#include "Aplication.h"
```

## Functions

- void [SysTick\\_Init](#) (void)  
: Initialize the systick
- void [SysTick\\_Off](#) (void)  
: Stops the systick
- void [SysTick\\_Set](#) (uint32\_t freq)  
: Set the counter as  $\text{freq} * 10\text{mS} - 1$
- void [SysTick\\_Handler](#) (void)  
: Interrupt each 10mS

## Variables

- uint32\_t [tick](#)  
Declared in main.

### 4.21.1 Detailed Description

: Firmware functions for SysTick

: Only develop for 30MHz

Author

: Tobias Bavasso Piizzi

Date

: 04/01/2021

### 4.21.2 Function Documentation

#### 4.21.2.1 SysTick\_Handler()

```
:void SysTick_Handler (  
    void )
```

: Interrupt each 10mS

: when the tick is out i know that happend time =  $\text{tick} * 10\text{mS}$

Author

: Tobias Bavasso Piizzi

Date

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 61 of file SysTick\_FW.c.

```
61         {  
62  
63         if (tick >= 0U)  
64             tick--;  
65  
66  
67  
68 }
```

**4.21.2.2 SysTick\_Init()**

```
:void SysTick_Init (  
    void )
```

: Initialize the systick

: Enable SysTick, enable interrupt and set the counter

**Author**

: Tobias Bavasso Piizzi

**Date**

: 04/01/2021

**Parameters**

|  |           |
|--|-----------|
|  | [in] void |
|--|-----------|

**Returns**

: void

Definition at line 19 of file SysTick\_FW.c.

```
19         {  
20     SysTick_Set(FRE30MHz);  
21     SYST_CSR = SYSTICK_ENABLE_INTERRUPT_CLK;  
22     SYST_CVR = 0;  
23 }
```

### 4.21.2.3 SysTick\_Off()

```
: SysTick_Off (
    void )
```

: Stops the systick

: disable SysTick, disable interrupt

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|           |
|-----------|
| [in] void |
|-----------|

#### Returns

: void

Definition at line 34 of file SysTick\_FW.c.

```
34         {
35     SYST_CSR = SYSTICK_DISABLE;
36 }
```

### 4.21.2.4 SysTick\_Set()

```
:void SysTick_Set (
    uint32_t freq )
```

: Set the counter as freq\*10mS -1

: Always use at 30MHz

#### Author

: Tobias Bavasso Piizzi

#### Date

: 04/01/2021

#### Parameters

|  |                              |
|--|------------------------------|
|  | [in] uint32_t freq: FRE30MHz |
|--|------------------------------|

#### Returns

: void

Definition at line 47 of file SysTick\_FW.c.

```
47      {  
48          SYST_RVR = freq*10 - 1; // 30MHz*10mS-1  
49      }
```

### 4.21.3 Variable Documentation

#### 4.21.3.1 tick

uint32\_t tick [extern]

Declared in main.

Declared in main.

Definition at line 20 of file 05-ConversionADC.c.

# Index

## 05-ConversionADC.c

main, [54](#)

tick, [54](#)

## ADC0\_SEQA\_IRQHandler

ADC\_FW.c, [56](#)

adc\_chan, [5](#)

adc\_ctrl\_t, [6](#)

## ADC\_Disable

ADC\_FW.c, [56](#)

ADC\_FW.h, [11](#)

## ADC\_Enable

ADC\_FW.c, [57](#)

ADC\_FW.h, [12](#)

## ADC\_FW.c

ADC0\_SEQA\_IRQHandler, [56](#)

ADC\_Disable, [56](#)

ADC\_Enable, [57](#)

ADC\_Init, [58](#)

ADC\_Power, [59](#)

## ADC\_FW.h

ADC\_Disable, [11](#)

ADC\_Enable, [12](#)

ADC\_Init, [12](#)

ADC\_Power, [14](#)

## ADC\_Init

ADC\_FW.c, [58](#)

ADC\_FW.h, [12](#)

## ADC\_Power

ADC\_FW.c, [59](#)

ADC\_FW.h, [14](#)

adc\_seqX\_ctrl, [7](#)

adc\_seqX\_gdat, [7](#)

adc\_thr, [8](#)

## Aplication.c

GPIO\_Init, [61](#)

LPC\_Init, [61](#)

tick, [62](#)

## Aplication.h

GPIO\_Init, [15](#)

LPC\_Init, [16](#)

## BoardClockRUN

SYSCON\_FW.c, [90](#)

SYSCON\_FW.h, [49](#)

## Digits\_to\_BCD7seg

Disp7Seg\_SW.c, [66](#)

## Disp7Seg\_FW.c

DISP7SEG\_Init, [63](#)

DISP\_Sweep, [64](#)

## Disp7Seg\_FW.h

DISP7SEG\_Init, [18](#)

DISP\_Sweep, [19](#)

## DISP7SEG\_Init

Disp7Seg\_FW.c, [63](#)

Disp7Seg\_FW.h, [18](#)

## Disp7Seg\_SW.c

Digits\_to\_BCD7seg, [66](#)

Display, [66](#)

## Disp7Seg\_SW.h

Display, [21](#)

## DISP\_Sweep

Disp7Seg\_FW.c, [64](#)

Disp7Seg\_FW.h, [19](#)

## Display

Disp7Seg\_SW.c, [66](#)

Disp7Seg\_SW.h, [21](#)

## GetInput

GPIO\_SW.c, [83](#)

GPIO\_SW.h, [38](#)

## GetOFFSET

GPIO\_FW.c, [68](#)

GPIO\_FW.h, [24](#)

## GetUserKEY

GPIO\_SW.c, [84](#)

GPIO\_SW.h, [39](#)

## GPIO\_ClearOUT

GPIO\_FW.c, [69](#)

GPIO\_FW.h, [24](#)

## GPIO\_Debounce

GPIO\_FW.c, [69](#)

GPIO\_FW.h, [25](#)

## GPIO\_DebounceUserKEY

GPIO\_FW.c, [70](#)

GPIO\_FW.h, [26](#)

## GPIO\_Disable

GPIO\_FW.c, [71](#)

GPIO\_FW.h, [27](#)

## GPIO\_Enable

GPIO\_FW.c, [72](#)

GPIO\_FW.h, [27](#)

## GPIO\_FW.c

GetOFFSET, [68](#)

GPIO\_ClearOUT, [69](#)

GPIO\_Debounce, [69](#)

GPIO\_DebounceUserKEY, [70](#)

GPIO\_Disable, [71](#)

GPIO\_Enable, [72](#)

- GPIO\_GetPIN, [72](#)
- GPIO\_SetDIR, [73](#)
- GPIO\_SetModeCLKDIV, [74](#)
- GPIO\_SetModeDAC, [74](#)
- GPIO\_SetModeFILTER, [75](#)
- GPIO\_SetModeHYS, [76](#)
- GPIO\_SetModel2C, [76](#)
- GPIO\_SetModelINPUT, [77](#)
- GPIO\_SetModelINV, [78](#)
- GPIO\_SetModeOD, [78](#)
- GPIO\_SetOUT, [79](#)
- GPIO\_SetPIN, [80](#)
- GPIO\_ToogleOUT, [80](#)
- IOCONDisable, [81](#)
- IOCONEnable, [82](#)
- offset, [82](#)
- GPIO\_FW.h
  - GetOFFSET, [24](#)
  - GPIO\_ClearOUT, [24](#)
  - GPIO\_Debounce, [25](#)
  - GPIO\_DebounceUserKEY, [26](#)
  - GPIO\_Disable, [27](#)
  - GPIO\_Enable, [27](#)
  - GPIO\_GetPIN, [28](#)
  - GPIO\_SetDIR, [29](#)
  - GPIO\_SetModeCLKDIV, [29](#)
  - GPIO\_SetModeDAC, [30](#)
  - GPIO\_SetModeFILTER, [31](#)
  - GPIO\_SetModeHYS, [31](#)
  - GPIO\_SetModel2C, [32](#)
  - GPIO\_SetModelINPUT, [33](#)
  - GPIO\_SetModelINV, [33](#)
  - GPIO\_SetModeOD, [34](#)
  - GPIO\_SetOUT, [35](#)
  - GPIO\_SetPIN, [35](#)
  - GPIO\_ToogleOUT, [36](#)
  - IOCONDisable, [37](#)
  - IOCONEnable, [37](#)
- GPIO\_GetPIN
  - GPIO\_FW.c, [72](#)
  - GPIO\_FW.h, [28](#)
- GPIO\_Init
  - Aplication.c, [61](#)
  - Aplication.h, [15](#)
- GPIO\_SetDIR
  - GPIO\_FW.c, [73](#)
  - GPIO\_FW.h, [29](#)
- GPIO\_SetModeCLKDIV
  - GPIO\_FW.c, [74](#)
  - GPIO\_FW.h, [29](#)
- GPIO\_SetModeDAC
  - GPIO\_FW.c, [74](#)
  - GPIO\_FW.h, [30](#)
- GPIO\_SetModeFILTER
  - GPIO\_FW.c, [75](#)
  - GPIO\_FW.h, [31](#)
- GPIO\_SetModeHYS
  - GPIO\_FW.c, [76](#)
- GPIO\_FW.h, [31](#)
- GPIO\_SetModel2C
  - GPIO\_FW.c, [76](#)
  - GPIO\_FW.h, [32](#)
- GPIO\_SetModelINPUT
  - GPIO\_FW.c, [77](#)
  - GPIO\_FW.h, [33](#)
- GPIO\_SetModelINV
  - GPIO\_FW.c, [78](#)
  - GPIO\_FW.h, [33](#)
- GPIO\_SetModeOD
  - GPIO\_FW.c, [78](#)
  - GPIO\_FW.h, [34](#)
- GPIO\_SetOUT
  - GPIO\_FW.c, [79](#)
  - GPIO\_FW.h, [35](#)
- GPIO\_SetPIN
  - GPIO\_FW.c, [80](#)
  - GPIO\_FW.h, [35](#)
- GPIO\_SW.c
  - GetInput, [83](#)
  - GetUserKEY, [84](#)
- GPIO\_SW.h
  - GetInput, [38](#)
  - GetUserKEY, [39](#)
- GPIO\_ToogleOUT
  - GPIO\_FW.c, [80](#)
  - GPIO\_FW.h, [36](#)
- inc/ADC\_FW.h, [9](#)
- inc/Aplication.h, [15](#)
- inc/Disp7Seg\_FW.h, [17](#)
- inc/Disp7Seg\_SW.h, [20](#)
- inc/GPIO\_FW.h, [21](#)
- inc/GPIO\_SW.h, [38](#)
- inc/LPC845.h, [40](#)
- inc/SwitchMatrix\_FW.h, [41](#)
- inc/SYSCON\_FW.h, [46](#)
- inc/SysTick\_FW.h, [50](#)
- IOCONDisable
  - GPIO\_FW.c, [81](#)
  - GPIO\_FW.h, [37](#)
- IOCONEnable
  - GPIO\_FW.c, [82](#)
  - GPIO\_FW.h, [37](#)
- LPC\_Init
  - Aplication.c, [61](#)
  - Aplication.h, [16](#)
- main
  - 05-ConversionADC.c, [54](#)
- offset
  - GPIO\_FW.c, [82](#)
- source/05-ConversionADC.c, [53](#)
- source/ADC\_FW.c, [55](#)
- source/Aplication.c, [60](#)



- source/Disp7Seg\_FW.c, [62](#)
- source/Disp7Seg\_SW.c, [65](#)
- source/GPIO\_FW.c, [67](#)
- source/GPIO\_SW.c, [83](#)
- source/mtb.c, [85](#)
- source/SwitchMatrix\_FW.c, [86](#)
- source/SYSCON\_FW.c, [89](#)
- source/SysTick\_FW.c, [90](#)
- SwitchMatrix\_FW.c
  - SWM, [86](#)
  - SWM\_Disable, [87](#)
  - SWM\_Enable, [87](#)
  - SWM\_PinEnable, [88](#)
- SwitchMatrix\_FW.h
  - SWM, [44](#)
  - SWM\_Disable, [44](#)
  - SWM\_Enable, [45](#)
  - SWM\_PinEnable, [46](#)
  - U0\_RXD, [42](#)
  - UO\_CTS, [43](#)
  - UO\_RTS, [43](#)
  - UO\_TXD, [42](#)
- SWM
  - SwitchMatrix\_FW.c, [86](#)
  - SwitchMatrix\_FW.h, [44](#)
- SWM\_Disable
  - SwitchMatrix\_FW.c, [87](#)
  - SwitchMatrix\_FW.h, [44](#)
- SWM\_Enable
  - SwitchMatrix\_FW.c, [87](#)
  - SwitchMatrix\_FW.h, [45](#)
- SWM\_PinEnable
  - SwitchMatrix\_FW.c, [88](#)
  - SwitchMatrix\_FW.h, [46](#)
- SYSCON\_FW.c
  - BoardClockRUN, [90](#)
- SYSCON\_FW.h
  - BoardClockRUN, [49](#)
- SysTick\_FW.c
  - SysTick\_Handler, [91](#)
  - SysTick\_Init, [92](#)
  - SysTick\_Off, [92](#)
  - SysTick\_Set, [93](#)
  - tick, [94](#)
- SysTick\_FW.h
  - SysTick\_Init, [51](#)
  - SysTick\_Off, [51](#)
  - SysTick\_Set, [52](#)
- SysTick\_Handler
  - SysTick\_FW.c, [91](#)
- SysTick\_Init
  - SysTick\_FW.c, [92](#)
  - SysTick\_FW.h, [51](#)
- SysTick\_Off
  - SysTick\_FW.c, [92](#)
  - SysTick\_FW.h, [51](#)
- SysTick\_Set
  - SysTick\_FW.c, [93](#)
- SysTick\_FW.h, [52](#)
- tick
  - 05-ConversionADC.c, [54](#)
  - Aplication.c, [62](#)
  - SysTick\_FW.c, [94](#)
- U0\_RXD
  - SwitchMatrix\_FW.h, [42](#)
- UO\_CTS
  - SwitchMatrix\_FW.h, [43](#)
- UO\_RTS
  - SwitchMatrix\_FW.h, [43](#)
- UO\_TXD
  - SwitchMatrix\_FW.h, [42](#)