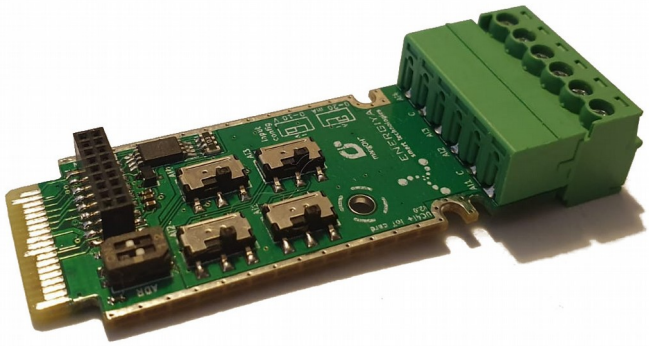
ENERGIYA

UNIVERSAL CONVERTER

SOFTWARE DOCUMENTATION



Typedefs

```
typedef struct le_ExtAdc_OnValueHandler* le_ExtAdc_OnValueHandlerRef_t;

typedef struct le_ExtAdc_OnAlertHandler* le_ExtAdc_OnAlertHandlerRef_t;

typedef void (*le_ExtAdc_HandlerRefFunc_t)(
      le_ExtAdc_IOT_ADC_t Sensor,
      uint16_t RawData,
      double Percentage,
      double ConvertedValue,
      void* contextPtr);

typedef void (*le_ExtAdc_AlertHandlerRefFunc_t)(
      le_ExtAdc_IOT_ADC_t Sensor,
      le_ExtAdc_Edges_t Edge,
      uint16_t RawData,
      double Percentage,
      double ConvertedValue,
      void* contextPtr);
```

Enumerations

```
enum le_ExtAdc_IOT_ADC_t {
      LE_EXTADC_IOT_ADC1 = 1,
      LE_EXTADC_IOT_ADC2 = 2,
      LE_EXTADC_IOT_ADC3 = 4,
      LE_EXTADC_IOT_ADC4 = 8}

enum le_ExtAdc_Edges_t {
      LE_EXTADC_NORMAL = 0,
      LE_EXTADC_TOOLOW = 1,
      LE_EXTADC_TOOHIGH = 2}

enum le_ExtAdc_Pins_t {
      LE_EXTADC_PIN00 = 0,
      LE_EXTADC_PIN01 = 1,
      LE_EXTADC_PIN10 = 2,
      LE_EXTADC_PIN11 = 3}

enum le_ExtAdc_InterfaceType_t {
      LE_EXTADC_MA_020 = 0,
      LE_EXTADC_MA_420 = 1,
      LE_EXTADC_V = 2}

enum le_ExtAdc_ConversionSpeed_t {
      LE_EXTADC_LOWPRECISION240SPS = 0,
      LE_EXTADC_HIGHPRECISION15SPS = 1}
```

Functions

```
le_result_t le_ExtAdc_SetDeviceAddress(le_ExtAdc_Pins_t Pin);

le_result_t le_ExtAdc_SetFileDescriptor(const char* Name);

le_result_t le_ExtAdc_GetValue(le_ExtAdc_IOT_ADC_t Sensor, uint16_t* RawDataPtr, double*
PercentagePtr, double* ConvertedValuePtr);

le_result_t le_ExtAdc_SetInterval(uint32_t Interval);

le_result_t le_ExtAdc_SetConverter(le_ExtAdc_IOT_ADC_t Sensor,double Min, double Max);

void le_ExtAdc_SetInterface(le_ExtAdc_IOT_ADC_t Sensor,le_ExtAdc_InterfaceType_t Interface);

le_result_t le_ExtAdc_SetEdges(le_ExtAdc_IOT_ADC_t Sensor, double TooLow, double TooHigh,
double Hysteresis);

void le_ExtAdc_MutexLock(void);

void le_ExtAdc_MutexUnlock(void);

le_ExtAdc_OnValueHandlerRef_t le_ExtAdc_AddOnValueHandler(uint8_t Sensor,
le_ExtAdc_HandlerRefFunc_t handlerPtr, void* contextPtr);

void le_ExtAdc_RemoveOnValueHandler(le_ExtAdc_OnValueHandlerRef_t addHandlerRef);

le_ExtAdc_OnAlertHandlerRef_t le_ExtAdc_AddOnAlertHandler(uint8_t Sensor,
le_ExtAdc_AlertHandlerRefFunc_t handlerPtr, void* contextPtr);

void le_ExtAdc_RemoveOnAlertHandler(le_ExtAdc_OnAlertHandlerRef_t addHandlerRef);

le_result_t le_ExtAdc_SetConversionSpeed(le_ExtAdc_ConversionSpeed_t Speed);
```

Detailed Description

Typedef Documentation

```
typedef struct le_ExtAdc_OnValueHandler* le_ExtAdc_OnValueHandlerRef_t;
```

Reference type used by Add/Remove functions for EVENT 'le_ExtAdc_OnValue'

```
typedef struct le_ExtAdc_OnAlertHandler* le_ExtAdc_OnAlertHandlerRef_t;
```

Reference type used by Add/Remove functions for EVENT 'le_ExtAdc_OnAlert'

```
typedef void (*le_ExtAdc_HandlerRefFunc_t)(
      le_ExtAdc_IOT_ADC_t Sensor,
      uint16_t RawData,
      double Percentage,
      double ConvertedValue,
      void* contextPtr);
```

Add handler function for EVENT 'le_ExtAdc_AddOnValueHandler'.
This event provides reading from specific sensor.
Reading will be returned every x number of milliseconds (Default 10000ms).
Default value can be changed by SetInterval(uint32 Interval).

Parameters

- [out] Sensor – Reading sensor. Enum value le_ExtAdc_IOT_ADC_t.
- [out] RawData – Raw data 15bit
- [out] Percentage – Reading in percent. Example If interface 0-10V is set up and measured value is 2V, parameter returns 20%.
- [out] ConvertedValue – Returns converted value - see **le_ExtAdc_SetConverter**
- [out] contextPtr

```
typedef void (*le_ExtAdc_AlertHandlerRefFunc_t)(
      le_ExtAdc_IOT_ADC_t Sensor,
      le_ExtAdc_Edges_t Edge,
      uint16_t RawData,
      double Percentage,
      double ConvertedValue,
      void* contextPtr);
```

Add handler function for EVENT 'le_ExtAdc_AddOnAlertHandler'.
This event fires when value is too low, too high or back to normal.
Edges are set by function **le_ExtAdc_SetEdges**.

Parameters

- [out] Sensor – Reading sensor. Enum value le_ExtAdc_IOT_ADC_t.
- [out] Edge – Notify if value is too low, too high or back to normal.
- [out] RawData – Raw data 15bit
- [out] Percentage – Reading in percent. Example If interface 0-10V is set up and measured value is 2V, parameter returns 20%.
- [out] ConvertedValue – Returns converted value - see **le_ExtAdc_SetConverter**
- [out] contextPtr

Enumeration Type Documentation

```
enum le_ExtAdc_IOT_ADC_t
```

Enumeration of available ADC sensors

Enumerator

LE_EXTADC_IOT_ADC1 sensor no. 1

LE_EXTADC_IOT_ADC2	sensor no. 2
LE_EXTADC_IOT_ADC3	sensor no. 3
LE_EXTADC_IOT_ADC4	sensor no. 4

---

enum le_ExtAdc_Edges_t

Enumeration of statuses.

Enumerator

LE_EXTADC_NORMAL	normal
LE_EXTADC_TOOLOW	too low
LE_EXTADC_TOOHIGH	too high

---

enum le_ExtAdc_Pins_t

Enumerator of all switch combinations. Hardware and software combination has to be the same. Combinations set up I2C addresses

Enumerator

LE_EXTADC_PIN00	when both ways are down - address 0x68 is set up.
LE_EXTADC_PIN01	when first is down, second is up - address 0x6C is set up.
LE_EXTADC_PIN10	when first is up, second is down - address 0x6A is set up.
LE_EXTADC_PIN11	when both ways are up - address 0x6E is set up.

---

enum le_ExtAdc_InterfaceType_t

Enumerator of interfaces

Enumerator

LE_EXTADC_MA_020	if interaface 0-20mA is used
LE_EXTADC_MA_420	if interaface 4-20mA is used
LE_EXTADC_V	if interaface 0-10V is used

---

enum le_ExtAdc_ConversionSpeed_t

Enumerator of conversion speeds

Enumerator

LE_EXTADC_LOWPRECISION240SPS	quick conversion 240 SPS with low precision (12 bits)
LE_EXTADC_HIGHPRECISION15SPS	slow conversion 15 SPS with high precision (16 bits)

Function Documentation

le_result_t **le_ExtAdc_SetDeviceAddress**(le_ExtAdc_Pins_t Pin)

Set device address. Device address has to be the same in the program and on IoT.
Hardware switch allows to setup address on IoT. Hardware combination must be passed to the program through the function.

Returns

LE_OK	Successful.
LE_BAD_PARAMETER	Bad parameter.

Parameters

[in] Pin	enum le_ExtAdc_Pins_t

---

le_result_t **le_ExtAdc_SetFileDescriptor**(**const char*** Name)

Set file descriptor.

Returns

LE_OK	Successful.
LE_BAD_PARAMETER	Bad parameter.
LE_FAULT	Error.

Parameters

[in] Name	File descriptor. Example "/dev/i2c-0"

---

le_result_t **le_ExtAdc_GetValue**(le_ExtAdc_IOT_ADC_t Sensor, **uint16_t*** RawDataPtr, **double*** PercentagePtr, **double*** ConvertedValuePtr)

Get the values of an ADC input from specific sensor.

Returns

LE_OK	Successful.
LE_UNDERFLOW	Raw value is below 6400 (when interface 4-20m is set)
LE_OVERFLOW	Raw value is above 32000.
LE_BAD_PARAMETER	Bad parameter.

Parameters

[in] Name	File descriptor. Example "/dev/i2c-0"

---

le_result_t **le_ExtAdc_SetInterval**(**uint32_t** Interval)

Set the time interval which elapses between two reading/checking, provided by events.

Returns

LE_OK	Successful.

Parameters

[in] Interval	Time in milisecond.

---

le_result_t le_ExtAdc_SetConverter(le_ExtAdc_IOT_ADC_t Sensor,**double** Min, **double** Max)

Function setup Min and Max achievable by connected unit.
Convertible value will be calculated based on the Min and Max.

Returns

LE_OK	Successful.
LE_BAD_PARAMETER	Bad parameter.

Parameters

[in] le_ExtAdc_IOT_ADC_t	Specific sensor.

```
      [in] Min                  Min value.
      [in] Max                  Max value.
```

```
void le_ExtAdc_SetInterface(le_ExtAdc_IOT_ADC_t Sensor,le_ExtAdc_InterfaceType_t Interface)
```

Set inteface on specific sensor.

Parameters

```
      [in] le_ExtAdc_IOT_ADC_t            Specific sensor.
      [in] le_ExtAdc_InterfaceType_t      Interface.
```

```
le_result_t le_ExtAdc_SetEdges(le_ExtAdc_IOT_ADC_t Sensor, double TooLow, double TooHigh,
double Hysteresis)
```

Set edges and hysteresis. Event OnAlert will be fired, based on these parameters.

Returns

```
      LE_OK             Successful.
      LE_BAD_PARAMETER  Bad parameter.
```

Parameters

```
      [in] le_ExtAdc_IOT_ADC_t  Specific sensor.
      [in] TooLow               Below this value system considers value as too low.
      [in] TooHigh              Above this value system considers value as too high.
```

```
void le_ExtAdc_MutexLock(void)
```

Function locks I2C communication in the component.
Should be used to avoid conflict with other I2C call.

```
void le_ExtAdc_MutexUnlock(void)
```

Function Unlocks I2C communication in the component.

```
le_ExtAdc_OnValueHandlerRef_t le_ExtAdc_AddOnValueHandler(uint8_t Sensor,
le_ExtAdc_HandlerRefFunc_t handlerPtr, void* contextPtr)
```

Add handler function for EVENT 'le_ExtAdc_OnValue'.
Register a callback function to be called in specific intervals.

Returns

```
      [out] HandlerRef
```

Parameters

```
      [in] Sensor                         Sensor to read can be combined. For example
                                          LE_EXTADC_IOT_ADC1|LE_EXTADC_IOT_ADC2.
      [in] le_ExtAdc_HandlerRefFunc_t     Handler to a function.
      [in] contextPtr
```

```
void le_ExtAdc_RemoveOnValueHandler(le_ExtAdc_OnValueHandlerRef_t addHandlerRef)
```

Remove handler function for EVENT 'le_ExtAdc_OnValue'

Parameters

[in] addHandlerRef

---

le_ExtAdc_OnAlertHandlerRef_t **le_ExtAdc_AddOnAlertHandler**(**uint8_t** Sensor,
le_ExtAdc_AlertHandlerRefFunc_t handlerPtr, **void*** contextPtr)

Add handler function for EVENT 'le_ExtAdc_OnAlert'
Register a callback function to be called when value is too low, too high or back to normal.

Returns
[out] HandlerRef

Parameters

[in] Sensor	Sensor to read can be combined. For example LE_EXTADC_IOT_ADC1|LE_EXTADC_IOT_ADC2.
[in] le_ExtAdc_AlertHandlerRefFunc_t	Handler to a function.
[in] contextPtr	

---

void le_ExtAdc_RemoveOnAlertHandler(le_ExtAdc_OnAlertHandlerRef_t addHandlerRef)

Remove handler function for EVENT 'le_ExtAdc_OnAlert'

Parameters
[in] addHandlerRef

---

le_result_t **le_ExtAdc_SetConversionSpeed**(le_ExtAdc_ConversionSpeed_t Speed)

Set conversion speed.

Returns

LE_OK	Successful.

Parameters
[in] Speed enum conversion speed.
