

openel.h File Reference

Macros

```
#define EL_TRUE 1
#define EL_FALSE 0
#define EL_NXT_PORT_A 0
#define EL_NXT_PORT_B 1
#define EL_NXT_PORT_C 2
#define EL_NXT_PORT_S1 0
#define EL_NXT_PORT_S2 1
#define EL_NXT_PORT_S3 2
#define EL_NXT_PORT_S4 4
#define EL_BT_NO_INIT 4
#define EL_BT_INITIALIZED 5
#define EL_BT_CONNECTED 6
#define EL_BT_STREAM 7
#define OPENEL_MAJOR 0
#define OPENEL_MINOR 1
#define OPENEL_PATHLEVEL 1
#define OPENEL_VERSION "OpenEL 0.1.1"
```

Typedefs

```
typedef signed char ELChar
typedef unsigned char ELUChar
typedef signed char ELInt8
typedef signed short ELInt16
typedef signed int ELInt32
typedef signed long long ELInt64
typedef unsigned char ELUInt8
typedef unsigned short ELUInt16
typedef unsigned int ELUInt32
typedef unsigned long long ELUInt64
typedef float ELFloat32
typedef double ELFloat64
typedef unsigned char ELBool
```

Functions

```
ELFloat64 elMotorGetAngle (ELUInt32 portid)
ELFloat64 elMotorSetAngle (ELUInt32 portid, ELFloat64 angle, ELInt32 speed)
void elMotorResetEncoder (ELUInt32 portid)
ELInt32 elMotorGetSpeed (ELUInt32 portid)
void elMotorSetSpeed (ELUInt32 portid, ELInt32 speed)
ELBool elMotorGetBrake (ELUInt32 portid)
void elMotorSetBrake (ELUInt32 portid, ELBool brake)
ELUInt16 elGyroSensorGetValue (ELUInt32 portid)
ELUInt16 elGyroSensorGetOffset (ELUInt32 portid)
void elGyroSensorSetOffset (ELUInt32 portid, ELUInt16 offset)
ELUInt16 elLightSensorGetValue (ELUInt32 portid)
ELBool elLightSensorGetLED (ELUInt32 portid)
void elLightSensorSetLED (ELUInt32 portid, ELBool light)
ELBool elTouchSensorGetState (ELUInt32 portid)
```

```
ELUInt16 elBatteryGetVoltage (void)
ELBool elSpeakerOutput (ELUInt32 freq, ELUInt32 ms, ELUInt32 vol)
void elSonarSensorInitialize (ELUInt32 portid)
void elSonarSensorTerminate (ELUInt32 portid)
ELInt32 elSonarSensorGetValue (ELUInt32 portid)
void elBluetoothInitializeMaster (const ELUChar *addr, const char *pin)
void elBluetoothInitializeSlave (const char *pin)
void elBluetoothTerminate (void)
ELUInt32 elBluetoothSendData (const void *buf, ELUInt32 offset, ELUInt32 len)
ELUInt32 elBluetoothReceiveData (void *buf, ELUInt32 offset, ELUInt32 len)
ELBool elBluetoothGetDeviceName (char *name)
ELBool elBluetoothSetDeviceName (const char *name)
ELInt32 elBluetoothGetStatus (void)
ELInt16 elBluetoothGetSignalStrength (void)
```

Macro Definition Documentation

```
#define EL_BT_CONNECTED 6
```

```
#define EL_BT_INITIALIZED 5
```

```
#define EL_BT_NO_INIT 4
```

```
#define EL_BT_STREAM 7
```

```
#define EL_FALSE 0
```

```
#define EL_NXT_PORT_A 0
```

```
#define EL_NXT_PORT_B 1
```

```
#define EL_NXT_PORT_C 2
```

```
#define EL_NXT_PORT_S1 0
```

```
#define EL_NXT_PORT_S2 1
```

```
#define EL_NXT_PORT_S3 2
```

```
#define EL_NXT_PORT_S4 4
```

```
#define EL_TRUE 1
```

```
#define OPENEL_MAJOR 0
```

```
#define OPENEL_MINOR 1
```

```
#define OPENEL_PATHLEVEL 1
```

```
#define OPENEL_VERSION "OpenEL 0.1.1"
```

Typedef Documentation

```
typedef unsigned char ELBool
```

```
typedef signed char ELChar
```

```
typedef float ELFloat32
```

```
typedef double ELFloat64
```

```
typedef signed short ELInt16
```

```
typedef signed int ELInt32
```

```
typedef signed long long ELInt64
```

```
typedef signed char ELInt8
```

```
typedef unsigned char ELUChar
```

```
typedef unsigned short ELUInt16
```

```
typedef unsigned int ELUInt32
```

```
typedef unsigned long long ELUInt64
```

```
typedef unsigned char ELUInt8
```

Function Documentation

```
ELUInt16 elBatteryGetVoltage ( void )
```

Gets the battery voltage.

Returns:
the current battery voltage.

```
ELBool elBluetoothGetDeviceName ( char * name )
```

Gets the Bluetooth device name.

Parameters:
[in] **name** the head address of the buffer that stores the device name.

Returns:
true if succeeded to get the device name, false otherwise.

```
ELInt16 elBluetoothGetSignalStrength ( void )
```

Gets the signal strength of the Bluetooth.
If the connection has not been established, it returns -1.

Returns:
the strength of the Bluetooth. (range: [0,100])

```
ELInt32 elBluetoothGetStatus ( void )
```

Gets the status of the connection of the Bluetooth.

List of constants representing the status of Bluetooth connection:

- **EL_BT_NO_INIT(Uninitialized state)**
- **EL_BT_INITIALIZED(Initialized state)**
- **EL_BT_CONNECTED(Connection established state)**
- **EL_BT_STREAM(State data can be transmitted and received)**

Returns:
the constant representing the status of Bluetooth connection.

```
void eBluetoothInitializeMaster ( const ELUChar * addr,
                                const char *   pin
                                )
```

Initializes the Bluetooth as a master device.

Parameters:

[in] **addr** the head address of the Bluetooth device address of a slave device.
[in] **pin** the head address of the pin code for the passkey exchange.

```
void eBluetoothInitializeSlave ( const char * pin )
```

Initializes the Bluetooth as a slave device.

Parameters:

[in] **pin** the head address of the pin code for the passkey exchange.

```
ELUInt32 eBluetoothReceiveData ( void *   buf,
                                 ELUInt32 offset,
                                 ELUInt32 len
                                 )
```

Receives the data to the buffer via the Bluetooth.

Parameters:

[in] **buf** the head address of the receiving data buffer.
[in] **offset** the offset of the receiving data buffer.
[in] **len** the receiving data buffer size.

Returns:

the number of bytes of data received.

```
ELUInt32 eBluetoothSendData ( const void * buf,
                              ELUInt32   offset,
                              ELUInt32   len
                              )
```

Sends the data in the buffer via the Bluetooth.

Parameters:

[in] **buf** the head address of the sending data buffer.
[in] **offset** the offset of the sending data buffer.
[in] **len** the sending data buffer size.

Returns:

the number of bytes of data sent.

```
ELBool eBluetoothSetDeviceName ( const char * name )
```

Sets the Bluetooth device name.

Parameters:

[in] **name** the device name.

Returns:

true if succeeded to set the device name, false otherwise.

```
void eBluetoothTerminate ( void )
```

Ternimates the Bluetooth.

This function can be used in both the master device and slave devices.

```
ELUInt16 eGyroSensorGetOffset ( ELUInt32 portid )
```

Gets the offset value of the gyro sensor.

Parameters:

[in] **portid** the port id of the gyro sensor

Returns:

the offset value of the gyro sensor.

```
ELUInt16 eGyroSensorGetValue ( ELUInt32 portid )
```

Gets the value of the gyro sensor.

Parameters:

[in] **portid** the port id of the gyro sensor.

Returns:

the value of the gyro sensor.

```
void eGyroSensorSetOffset ( ELUInt32 portid,
                            ELUInt16 offset
                            )
```

Sets the offset value of the gyro sensor.

Parameters:

[in] **portid** the port id of the gyro sensor.
[in] **offset** the offset value which is set to the gyro sensor.

ELBool eLightSensorGetLED (ELUInt32 portid)

Gets whether the LED of light sensor is turned on.

Parameters:

[in] **portid** the port id of the light sensor.

Returns:

true if the LED is turned on, false otherwise.

ELUInt16 eLightSensorGetValue (ELUInt32 portid)

Gets the value of the light sensor.

Parameters:

[in] **portid** the port id of the light sensor.

Returns:

the value of the light sensor.

```
void eLightSensorSetLED ( ELUInt32 portid,  
                          ELBool light  
                          )
```

Sets the lighting state of the LED of light sensor.

Parameters:

[in] **portid** the port id of the light sensor.

[in] **light** true if turn on the LED of light sensor, false otherwise.

ELFloat64 eMotorGetAngle (ELUInt32 portid)

Gets the angle of the encoder of the motor.

Parameters:

[in] **portid** the port id of the motor.

Returns:

the angle of the encoder. (unit: radian)

ELBool eMotorGetBrake (ELUInt32 portid)

Gets whether the brake of motor is enabled.

Parameters:

[in] **portid** the port id of the motor.

Returns:

true if the brake of motor is enabled, false otherwise.

ELInt32 eMotorGetSpeed (ELUInt32 portid)

Gets the rotational velocity(PWM value) which is set to the motor.

Parameters:

[in] **portid** the port id of the motor.

Returns:

the rotational velocity which is set to the motor. (range: [-100,100])

void eMotorResetEncoder (ELUInt32 portid)

Resets the encoder value, and set the current angle as a criteria(zero radian).

Parameters:

[in] **portid** the port id of the motor.

```
ELFloat64 eMotorSetAngle ( ELUInt32 portid,  
                           ELFloat64 angle,  
                           ELInt32 speed  
                           )
```

Rotates the motor to the specified angle.

If it is unable to do so, this function is finished.

A motor angle is defined as base angle(0radian) at the time of starting program or doing eMotorResetEncoder.

If This return value is difference between the parameter angle and the actual rotation angle.

Parameters:

[in] **portid** the port id of the motor.

[in] **angle** the angle specifies to the encoder. (unit: radian)

[in] **speed** the pwm value specifies to the motor. (range: [-100,100])

Returns:

the difference between the parameter angle and the actual rotation angle (unit: radian)

```
void eMotorSetBrake ( ELUInt32 portid,  
                    ELBool brake  
                    )
```

Enables/Disables the brake of motor.

Parameters:

[in] **portid** the port id of the motor.

[in] **brake** true enables the brake of motor, and false disables.

```
void eMotorSetSpeed ( ELUInt32 portid,  
                    ELInt32 speed  
                    )
```

Sets the rotational velocity(PWM value) to the motor.

Parameters:

[in] **portid** the port id of the motor.

[in] **speed** the rotational velocity which is set to the motor. (range: [-100,100])

```
ELInt32 eSonarSensorGetValue ( ELUInt32 portid )
```

Gets the value of sonar sensor.
If the sonar sensor has not been initialized, it returns -1.

Parameters:

[in] **portid** the port id of the sonar sensor.

Returns:

the measured distance. (unit: cm)

```
void eSonarSensorInitialize ( ELUInt32 portid )
```

Initializes the sonar sensor.
This function should be called only once before using the sonar sensor.

Parameters:

[in] **portid** the port id of the sonar sensor.

```
void eSonarSensorTerminate ( ELUInt32 portid )
```

Terminates the sonar sensor.
This function should be called only once before terminating the use of the sonar sensor.

Parameters:

[in] **portid** the port id of the sonar sensor.

```
ELBool eSpeakerOutput ( ELUInt32 freq,  
                      ELUInt32 ms,  
                      ELUInt32 vol  
                      )
```

Outputs the tone sound from the speaker.

Parameters:

[in] **freq** the frequency. (unit: Hz)

[in] **ms** the output duration. (unit: 10ms)

[in] **vol** the volume. (unit: %)

Returns:

true if the output was succeeded, false otherwise.

```
ELBool eTouchSensorGetState ( ELUInt32 portid )
```

Gets the touch sensor state.

Parameters:

[in] **portid** the port id of the touch sensor.

Returns:

true if the touch sensor is on, false otherwise.