BANO node API

texane@gmail.com

1 Overview

This document describes the BANO node application programming interface. It is intended for node developers.

BANO offers a runtime and the corresponding programming interface that abstracts a node application logic from low level details such as hardware architecture and protocol implementation. The interface is mainly descriptive and event based: the developer first initializes node related information. The BANO runtime then calls application handlers whenever appropriate: network messages reception, timers, hardware related interrupts ...

2 Reference

2.1 Files

common/bano_common.h: constants and types common to base and node
node/bano_node.h: function and type declarations
node/bano_node.c: function implementations

2.2 Types

```
typedef struct
{
  /* 100 milliseconds units, max 10736 */
  uint16_t timer_100ms;

  /* waking event mask */
#define BANO_WAKE_NONE 0
#define BANO_WAKE_TIMER (1 << 0)
#define BANO_WAKE_MSG (1 << 1)
#define BANO_WAKE_POLL (1 << 2)
#define BANO_WAKE_PCINT (1 << 3)
  uint8_t wake_mask;

  /* module disabling mask */
#define BANO_DISABLE_ADC (1 << 0)
#define BANO_DISABLE_WDT (1 << 1)
#define BANO_DISABLE_CMP (1 << 2)
#define BANO_DISABLE_USART (1 << 3)
#define BANO_DISABLE_NONE 0x00
#define BANO_DISABLE_ALL 0xff
  uint8_t disable_mask;

  uint32_t pcint_mask;

  /* NODL identifier */
  uint32_t nodl_id;

} bano_info_t;

static const bano_info_t bano_info_default =
{
  .wake_mask = BANO_WAKE_NONE,
  .disable_mask = BANO_DISABLE_ALL,
  .nodl_id = 0
};
```

2.3 Functions

```
/* exported by the runtime */
uint8_t bano_init(const bano_info_t*);
uint8_t bano_fini(void);
uint8_t bano_send_set(uint16_t, uint32_t);
uint8_t bano_wait_event(bano_msg_t*);
uint8_t bano_loop(void);

/* implemented by the application */
extern uint8_t bano_set_handler(uint16_t, uint32_t);
extern uint8_t bano_get_handler(uint16_t, uint32_t*);
extern uint8_t bano_timer_handler(void);
extern uint8_t bano_pcint_handler(void);
```

3 Example

```
/* example: enable disable a LED on BANO messages */

#include "bano/src/common/bano_common.h"
#include "bano/src/node/bano_node.h"
#include "bano/src/node/bano_node.c"


/* led routines */

#define LED_DDR DDRB
#define LED_PORT PORTB
#define LED_MASK (1 << 1)

static void led_set_high(void)
{
  LED_DDR |= LED_MASK;
  LED_PORT |= LED_MASK;
}

static void led_set_low(void)
{
  LED_DDR |= LED_MASK;
  LED_PORT &= ~LED_MASK;
}


/* event handlers */

#define LED_KEY 0x0000

static uint8_t led_value = 0;

uint8_t bano_get_handler(uint16_t key, uint32_t* val)
{
  /* called by the runtime on GET messages */

  if (key != LED_KEY) return (uint8_t)-1;
  *val = led_value;
  return 0;
}

uint8_t bano_set_handler(uint16_t key, uint32_t val)
{
  /* called by the runtime on SET messages */

  if (key != LED_KEY) return (uint8_t)-1;
  led_value = val;
  if (led_value == 0) led_set_low();
  else led_set_high();
  return 0;
}

int main(void)
{
  /* initialize the runtime and loop forever */

  bano_info_t info;

  info = bano_info_default;
  info.wake_mask |= BANO_WAKE_MSG;
  info.nodl_id = 0xdeadbeef;
  bano_init(&info);

  bano_loop();

  bano_fini();

  return 0;
}
```