GPIO Register Access

Functions:

- int bcm2835_i2c_begin (void)
- void bcm2835_i2c_end (void)
- void bcm2835_i2c_setSlaveAddress (uint8_t addr)
- void bcm2835_i2c_setClockDivider (uint16_t divider)
- void bcm2835_i2c_set_baudrate (uint32_t baudrate)
- uint8_t bcm2835_i2c_write (const char *buf, uint32_t len)
- uint8_t bcm2835_i2c_read (char *buf, uint32_t len)
- uint8_t bcm2835_i2c_read_register_rs (char *regaddr, char *buf, uint32_t len)
- uint8_t bcm2835_i2c_write_read_rs (char *cmds, uint32_t cmds_len, char *buf, uint32_t buf_len)

Detailed Description:

These functions let you use I2C (The Broadcom Serial Control bus with the Philips I2C bus/interface version 2.1 January 2000.) to interface with an external I2C device.

Function Documentation:

◆ bcm2835_i2c_begin()

int bcm2835_i2c_begin (void)

Start I2C operations. Forces RPi I2C pins P1-03 (SDA) and P1-05 (SCL) to alternate function ALT0, which enables those pins for I2C interface. You should call bcm2835_i2c_end() when all I2C functions are complete to return the pins to their default functions, it Returns 1 if successful, 0 otherwise (perhaps because you are not running as root)

◆ bcm2835_i2c_end()

void bcm2835_i2c_end (void)

End I2C operations. I2C pins P1-03 (SDA) and P1-05 (SCL) are returned to their default INPUT behaviour.

◆ bcm2835_i2c_read()

uint8_t bcm2835_i2c_read(char * buf, uint32_t len)

Transfers any number of bytes from the currently selected I2C slave. (as previously set by bcm2835_i2c_setSlaveAddress)

Parameters

- buf Buffer of bytes to receive.
- len Number of bytes in the buf buffer, and the number of bytes to received.

◆ bcm2835_i2c_read_register_rs()

uint8_t bcm2835_i2c_read_register_rs (char * regaddr, char * buf, uint32_t len)

Allows reading from I2C slaves that require a repeated start (without any prior stop) to read after the required slave register has been set. For example, the popular MPL3115A2 pressure and temperature sensor. Note that your device must support or require this mode. If your device does not require this mode then the standard combined:
bcm2835_i2c_write
bcm2835_i2c_read are a better choice. Will read from the slave previously set by
bcm2835_i2c_setSlaveAddress

Parameters

- regaddr Buffer containing the slave register you wish to read from.
- buf Buffer of bytes to receive.
- len Number of bytes in the buf buffer, and the number of bytes to received.

◆ bcm2835_i2c_set_baudrate()

void bcm2835_i2c_set_baudrate(uint32_t baudrate)

Sets the I2C clock divider by converting the baudrate parameter to the equivalent I2C clock divider.

For the I2C standard 100khz you would set baudrate to 100000 The use of baudrate corresponds to its use in the I2C kernel device driver.

◆ bcm2835_i2c_setClockDivider()
void bcm2835_i2c_setClockDivider (uint16_t divider)
Sets the I2C clock divider and therefore the I2C clock speed.
Parameters:

- divider: The desired I2C clock divider, one of BCM2835_I2C_CLOCK_DIVIDER_*, see bcm2835I2CClockDivider

◆ bcm2835_i2c_setSlaveAddress()
void bcm2835_i2c_setSlaveAddress (uint8_t addr)
Sets the I2C slave address.
Parameters

- addr: The I2C slave address.

◆ bcm2835_i2c_write()
uint8_t bcm2835_i2c_write (const char * buf, uint32_t len)
Transfers any number of bytes to the currently selected I2C slave.
Parameters

- buf: Buffer of bytes to send.
- Len: Number of bytes in the buf buffer, and the number of bytes to send.

◆ bcm2835_i2c_write_read_rs()
uint8_t bcm2835_i2c_write_read_rs (char * cmds, uint32_t cmds_len, char * buf, uint32_t buf_len)

Allows sending an arbitrary number of bytes to I2C slaves before issuing a repeated start (with no prior stop) and reading a response. Necessary for devices that require such behavior, such as the MLX90620. Will write to and read from the slave previously set by
Parameters

- cmds Buffer containing the bytes to send before the repeated start condition.
- cmds_len Number of bytes to send from cmds buffer
- buf Buffer of bytes to receive.
- buf_len Number of bytes to receive in the buf buffer.

_____