```
        /* Gas heater related coefficients */
        dev->calib.par_gh1 = (int8_t) coeff_array[BME680_GH1_REG];
        dev->calib.par_gh2 = (int16_t) (BME680_CONCAT_BYTES(coeff_array[BME680_GH2_MSB_REG],
            coeff_array[BME680_GH2_LSB_REG]));
        dev->calib.par_gh3 = (int8_t) coeff_array[BME680_GH3_REG];

        /* Other coefficients */
        if (rslt == BME680_OK) {
            rslt = bme680_get_regs(BME680_ADDR_RES_HEAT_RANGE_ADDR, &temp_var, 1, dev);

            dev->calib.res_heat_range = ((temp_var & BME680_RHRANGE_MSK) / 16);
            if (rslt == BME680_OK) {
                rslt = bme680_get_regs(BME680_ADDR_RES_HEAT_VAL_ADDR, &temp_var, 1, dev);

                dev->calib.res_heat_val = (int8_t) temp_var;
                if (rslt == BME680_OK)
                    rslt = bme680_get_regs(BME680_ADDR_RANGE_SW_ERR_ADDR, &temp_var, 1, dev);
            }
        }
        dev->calib.range_sw_err = ((int8_t) temp_var & (int8_t) BME680_RSERROR_MSK) / 16;
    }

    return rslt;
}

/*!
 * @brief This internal API is used to set the gas configuration of the sensor.
 */
static int8_t set_gas_config(struct bme680_dev *dev)
{
    int8_t rslt;

    /* Check for null pointer in the device structure*/
    rslt = null_ptr_check(dev);
    if (rslt == BME680_OK) {
```