Interfacing ADXL345 with ESP32 using ESP-IDF (I2C Protocol)

Introduction

This guide provides a step-by-step walkthrough for interfacing the ADXL345 3-axis accelerometer with the ESP32 microcontroller using the ESP-IDF framework via the I2C protocol. It covers hardware wiring, I2C setup, ADXL345 configuration, and reading raw acceleration data (X, Y, Z axes).

1. Hardware Connections (I2C)

Connect the ADXL345 to the ESP32 as follows:

ADXL345 Pin	ESP32 Pin
VCC	3.3V
GND	GND
SDA	GPIO21
SCL	GPIO22
CS	3.3V
SDO	GND or 3.3V (affects I2C address)

2. I2C Initialization (ESP-IDF)

Use the following function to configure ESP32 as an I2C master:

```
#define I2C_MASTER_NUM          I2C_NUM_0
#define I2C_MASTER_FREQ_HZ       100000
#define I2C_MASTER_SDA_IO       21
#define I2C_MASTER_SCL_IO       22

void i2c_master_init(void) {
    i2c_config_t conf = {
        .mode = I2C_MODE_MASTER,
        .sda_io_num = I2C_MASTER_SDA_IO,
        .scl_io_num = I2C_MASTER_SCL_IO,
        .sda_pullup_en = GPIO_PULLUP_ENABLE,
        .scl_pullup_en = GPIO_PULLUP_ENABLE,
        .master.clk_speed = I2C_MASTER_FREQ_HZ,
    };
    ESP_ERROR_CHECK(i2c_param_config(I2C_MASTER_NUM, &conf));
    ESP_ERROR_CHECK(i2c_driver_install(I2C_MASTER_NUM,
conf.mode, 0, 0, 0));
}
```

---

3. ADXL345 I2C Address

The default I2C address of the ADXL345 is 0x53 when the SDO pin is connected to GND.

If SDO is connected to 3.3V, the address becomes 0x1D.

Use an I2C scanner tool to confirm this before proceeding.

4. Register Configuration

Configure the following registers to initialize the ADXL345:

Register Name	Address	Purpose	Value
DEVID	0x00	Read-only, should return 0xE5	(read)
DATA_FORMAT	0x31	±2g full resolution	0x08
POWER_CTL	0x2D	Enable measurement mode	0x08
DATAX0	0x32	Start of acceleration data (X0)	-

To enable the sensor:

- - Write 0x08 to POWER_CTL (0x2D)
- - Write 0x08 to DATA_FORMAT (0x31) for ±2g range with full-resolution

5. Reading X, Y, Z Acceleration Data

To read 10-bit acceleration values from all axes:

```
void adxl345_read_xyz(int16_t *x, int16_t *y, int16_t *z) {
    uint8_t data[6];
    adxl345_read(0x32, data, 6);
    *x = (int16_t)((data[1] << 8) | data[0]);
    *y = (int16_t)((data[3] << 8) | data[2]);
    *z = (int16_t)((data[5] << 8) | data[4]);
}
```

This reads 6 bytes from the sensor: DATAX0 to DATAZ1.

Each axis returns a 16-bit signed integer (two's complement).

Convert raw values to g-force by:

```
acceleration_g = raw / 256.0 (if range is ±2g)
```

6. Summary of Steps

- Wire ADXL345 to ESP32 using I2C.
- Initialize I2C with ESP-IDF APIs.
- (Optional) Use an I2C scanner to verify device address (usually 0x53).
- Read and confirm device ID at register 0x00 (expect 0xE5).
- Write to DATA_FORMAT and POWER_CTL registers.
- Read and interpret 6 bytes starting from register 0x32 to get X/Y/Z acceleration.