IOT BASED NOISE POLLUTION MONITORING

To create a low-power IoT noise pollution monitoring system using a 300mAh battery as the power source, you will need to carefully design the circuit for power efficiency. Here's a circuit design and a MicroPython program for an ESP8266-based system:

Hardware Setup :

1. ESP8266 (NodeMCU): The ESP8266 will be the main microcontroller.

2. Noise Sensor: Connect the noise sensor to an analog input pin of the ESP8266. Make sure to check the sensor's datasheet for wiring details.

3. Battery: Connect the 300mAh battery to the ESP8266 through a voltage regulator to provide a stable 3.3V power supply. This regulator will help you reduce power consumption by maintaining a stable voltage even as the battery discharges.

4. Deep Sleep Mode: Configure the ESP8266 to enter deep sleep mode between measurements. During deep sleep, the ESP8266 consumes very little power.

MicroPython Program :

```
import machine
import time
import network
import ujson
import ubinascii
from umqtt.simple import MQTTClient
from machine import ADC, Pin
```

```
# Wi-Fi configuration
WIFI_SSID = "NoiseDet"
WIFI_PASSWORD = "Noise903Phase$"

# MQTT configuration
MQTT_BROKER = "mqtt.googleapis.com"
MQTT_TOPIC = "noise_data"

# Create a unique MQTT client ID based on the device's MAC address
client_id = ubinascii.hexlify(machine.unique_id()).decode()

# Initialize the ADC to read analog values from the noise sensor
adc = ADC(0)

# Set up a pin to control deep sleep
DEEP_SLEEP_PIN = Pin(16, Pin.IN, Pin.PULL_UP)  # GPIO 16

# Function to connect to Wi-Fi
def connect_wifi():
    sta_if = network.WLAN(network.STA_IF)
    if not sta_if.isconnected():
        sta_if.active(True)
        sta_if.connect(WIFI_SSID, WIFI_PASSWORD)
        while not sta_if.isconnected():
            pass

# Function to send data to MQTT broker
def send_data(data):
    client = MQTTClient(client_id, MQTT_BROKER)
    client.connect()
    client.publish(MQTT_TOPIC, data)
    client.disconnect()

# Main loop
while True:
    if DEEP_SLEEP_PIN.value() == 0:
        # If the deep sleep pin is held low, enter deep sleep mode
        machine.deepsleep()

    # Connect to Wi-Fi
    connect_wifi()
```

```
    noise_level = adc.read()  # Read noise level from the sensor
    data = {"noise_level": noise_level}
    payload = ujson.dumps(data)

    # Send the noise level data to the MQTT broker
    send_data(payload)

    # Disconnect from Wi-Fi to save power
    network.WLAN(network.STA_IF).active(False)

    # Enter deep sleep for a specified period (e.g., 10 minutes)
    machine.deepsleep(600000)
```

The program enters deep sleep mode when a certain condition is met (e.g., when a button connected to GPIO 16 is pressed). While in deep sleep, the ESP8266 consumes very little power, and it wakes up periodically to take measurements and send data.