

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**  
**KHOA CƠ KHÍ CHẾ TẠO MÁY**  
**BỘ MÔN CƠ ĐIỆN TỬ**



**HCMUTE**  
**TÀI LIỆU HƯỚNG DẪN SỬ DỤNG**  
**DRIVER ADS1115**

**GVHD:** TS. Bùi Hà Đức

**SVTH:** Hà Nhật Quang 21146502

Bạch Trần Anh Kiệt 22146159

Bùi Nguyễn Anh Khoa 22146149

**Tp. Hồ Chí Minh, tháng 5 năm 2025**

## 1. Giới thiệu sơ bộ về Driver:

Tên driver: ADS1115

Chức năng: Cho phép người dùng cấu hình, giao tiếp và thu thập dữ liệu từ bộ chuyển đổi ADC 16-bit ADS1115 thông qua giao tiếp I2C.

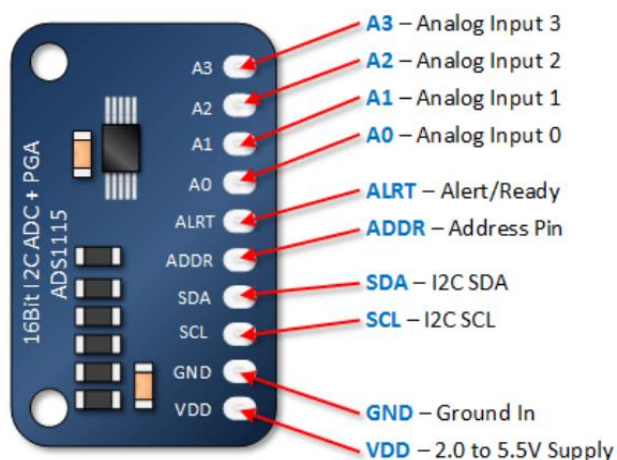
Nền tảng hỗ trợ: Raspberry Pi hoặc các hệ thống Linux có hỗ trợ I2C.

## 2. Giới thiệu sơ bộ về thiết bị:

Mạch chuyển tín hiệu ADC ADS1115 16-Bit 4-Channel I2C được sử dụng để tạo ra 4 kênh ADC ( Analog to Digital Converter) độ phân giải 16-bit giao tiếp với Vi điều khiển hoặc Máy tính nhúng (Raspberry Pi) qua giao tiếp I2C đơn giản với chỉ 2 chân tín hiệu (Data, Clock), mạch được ứng dụng để đọc tín hiệu từ các Module hoặc Cảm biến nhận tín hiệu Analog với độ chính xác cao hoặc thêm các chân ADC cho các mạch xử lý chỉ có chân tín hiệu Digital.

### Thông số kỹ thuật:

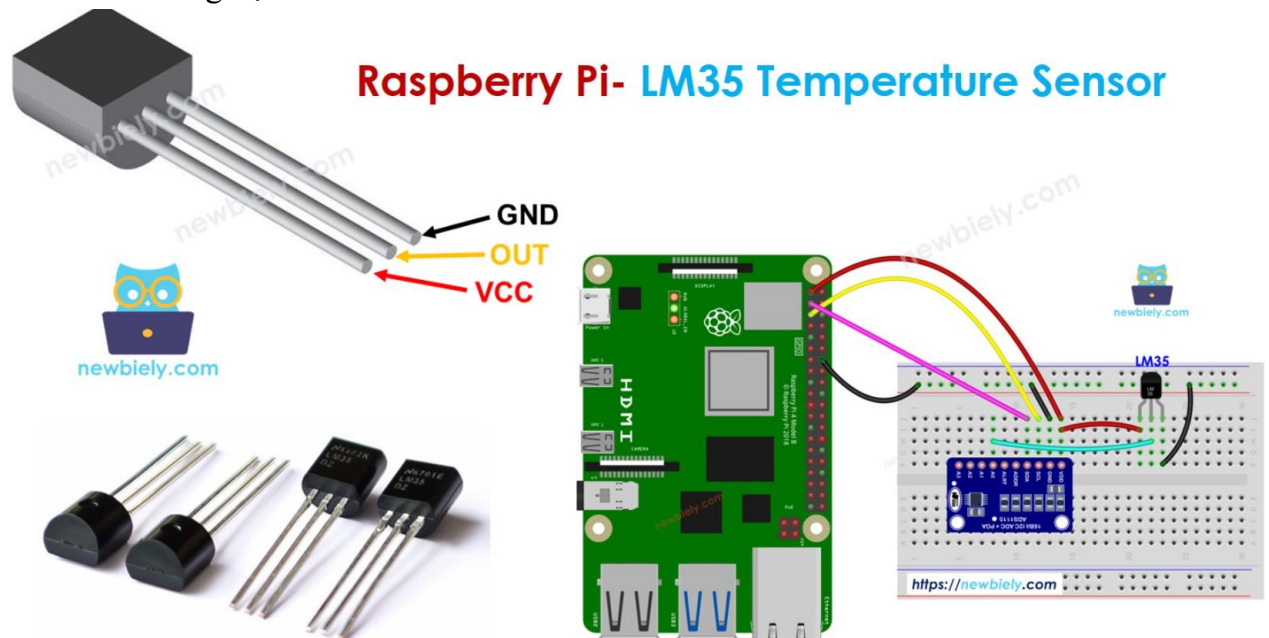
- IC chính: ADS1115 16-Bit ADC 4 Channel with Programmable Gain Amplifier
- Điện áp sử dụng: 2.0~5.5VDC
- Dòng tiêu thụ thấp:
  - Continuous Mode: Only 150 $\mu$ A
  - Single-Shot Mode: Auto Shut-Down
- Chuẩn giao tiếp: I2C (có thể thiết đặt địa chỉ qua 4 chân Set địa chỉ A0~A3)
- Có thể cấu hình tần số lấy mẫu: 8SPS~860SPS (Sample per Second)
- Internal LOW-DRIFT Voltage Reference
- Internal Oscillator
- Internal PGA
- I2C INTERFACE: Pin-Selectable Addresses
- Programmable Comparator



ADS1115 16-Bit ADC 4 Channel

### 3. Cách kết nối thiết bị với Raspberry:

Ở trường hợp này, chúng ta sẽ sử dụng cảm biến nhiệt độ LM35 như là một tín hiệu ADC để thử nghiệm driver ADS1115.



### 4. Cài đặt driver ADS1115:

#### 4.1. Kiểm tra thiết bị đã được kết nối với Raspberry chưa:

Sử dụng câu lệnh sau để kiểm tra bus I2C số 1:

+) `i2cdetect -y 1`

Nếu kết nối chính xác giữa Raspberry và thiết bị ADS1115 thì chúng ta sẽ nhận được bảng như sau:

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00:																
10:																
20:																
30:																
40:															48	
50:																
60:																
70:																

#### 4.2. Cài đặt driver:

Với cách này, yêu cầu phiên bản Linux version 5 để hạn chế lỗi nhất có thể.

Trước tiên, bạn cần tạo file overlay để thêm thiết bị ADS1115 vào bus I2C.

Dưới đây là nội dung mẫu:

```

/dts-v1/;
/plugin/;
&{/} {
    fragment@0 {
        target = <&i2c1>;
        __overlay__ {
            ads1115@48 {
                compatible = "TexasInstrument,ADS11150";
                reg = <0x48>;
                status = "okay";
            };
        };
    };
};
};
};

```

Lưu lại với tên ADS1115\_overlay.dts.

**Tiếp theo, biên dịch file .dts sang file .dtbo với câu lệnh sau:**

+) sudo dtc -@ -I dts -O dtb -o ADS1115\_overlay.dtbo ADS1115\_overlay.dts

**Tiếp theo, sao chép file .dtbo vào thư mục overlay của hệ thống:**

+) sudo cp ADS1115\_overlay.dtbo /boot/overlays/

**Tiếp theo, thêm overlay vào cấu hình boot:**

Mở file cấu hình boot: bash sudo nano /boot/config.txt

Thêm dòng sau vào cuối file: txt dtoverlay=ADS1115\_overlay

Lưu và khởi động lại hệ thống: sudo reboot

**Sau khi đã khởi động lại hệ thống, chuẩn bị một make file:**

Make file mẫu:

```
obj-m += ads1115_driver.o
```

all:

```
make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
```

clean:

```
make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

**Tiếp theo, biên dịch driver:**

+) make

Sau khi biên dịch thành công, sẽ sinh ra file ads1115\_driver.ko

**Tiếp theo, nạp module vào kernel:**

+) sudo insmod ads1115\_driver.ko

Kiểm tra bằng câu lệnh dmesg, nếu thành công thì sẽ có dòng ADS1115 driver installed successful.

## **5. Chương trình người dùng:**

### **5.1. Define tên thiết bị và mã ioctl:**

```
#define ADS1115_DEVICE "/dev/ADS1115"  
#define ADS1115_IOCTL_READ_CONFIG_IOWR('a', 1, struct  
ADS1115_read_config)
```

+) /dev/ADS1115 là node thiết bị tạo bởi driver.

+) ADS1115\_IOCTL\_READ\_CONFIG là mã ioctl do bạn định nghĩa trong driver để truyền nhận dữ liệu dạng struct ADS1115\_read\_config.

### **5.2. Cấu trúc dữ liệu gửi đi và nhận về:**

```
struct ADS1115_read_config {  
    uint8_t channel;
```

**+) MUX (Kênh đầu vào - channel):**

Người dùng có thể chọn kênh ADC bằng cách đặt cfg.channel như sau:

- 0: AIN0 vs GND
- 1: AIN1 vs GND

- 2: AIN2 vs GND
- 3: AIN3 vs GND
- 4: AIN0 – AIN1 (đo vi sai)
- 5: AIN0 – AIN3 (đo vi sai)
- 6: AIN1 – AIN3 (đo vi sai)
- 7: AIN2 – AIN3 (đo vi sai)

Tùy theo driver ánh xạ, người dùng chỉ cần chọn số kênh từ 0–7 mà không cần biết mã MUX cụ thể.

```
uint16_t pga;
```

#### **+) PGA (Programmable Gain Amplifier):**

Người dùng có thể chọn mức điện áp tối đa đầu vào bằng cách đặt `cfg.pga` với một trong các giá trị sau:

- 0x0000:  $\pm 6.144\text{V}$
- 0x0200:  $\pm 4.096\text{V}$
- 0x0400:  $\pm 2.048\text{V}$  (mặc định – phù hợp với LM35)
- 0x0600:  $\pm 1.024\text{V}$
- 0x0800:  $\pm 0.512\text{V}$
- 0x0A00:  $\pm 0.256\text{V}$

Điện áp đo được sẽ được chia theo hệ số tương ứng với phạm vi đã chọn.

```
uint16_t mode;
```

#### **+) Mode (Chế độ hoạt động):**

Người dùng có thể chọn chế độ hoạt động của ADC bằng cách đặt `cfg.mode` với một trong các giá trị sau:

- 0x0000: **Continuous mode** (liên tục)
- 0x0100: **Single-shot mode** (từng mẫu một)
- **Continuous mode** phù hợp với đọc liên tục (ví dụ đọc cảm biến nhiệt độ).
- **Single-shot mode** phù hợp khi chỉ cần lấy mẫu từng lần (ví dụ đo một giá trị rồi ngắt).

```
uint16_t datarate;
```

#### **+) Datarate (Tốc độ lấy mẫu):**

Người dùng có thể chọn tốc độ lấy mẫu ADC bằng cách đặt `cfg.datarate` với một trong các giá trị sau:

- 0x0000: 8 SPS
- 0x0020: 16 SPS
- 0x0040: 32 SPS
- 0x0060: 64 SPS
- 0x0080: 128 SPS (**mặc định – cân bằng giữa nhiễu và tốc độ**)
- 0x00A0: 250 SPS
- 0x00C0: 475 SPS
- 0x00E0: 860 SPS

Tốc độ càng cao thì độ trễ càng thấp nhưng có thể nhiễu nhiều hơn.

```
int16_t result;
};
```

### 5.3. **Hàm main: mở thiết bị, cấu hình, vòng đọc dữ liệu:**

```
fd = open(ADS1115_DEVICE, O_RDWR);
```

+) Mở file /dev/ADS1115, nếu lỗi thì in ra và thoát.

+) Thiết lập và gửi ngưỡng cảnh báo

```
th.low_threshold = 1000; // ví dụ: giá trị raw ADC
```

```
th.high_threshold = 20000; // ví dụ: giá trị raw ADC
```

```
if (ioctl(fd, ADS1115_IOCTL_SET_THRESHOLDS, &th) < 0) {
    perror("IOCTL set thresholds failed");
    close(fd);
    return -1;
} else {
    printf("Set thresholds: LOW=%d, HIGH=%d\n", th.low_threshold,
th.high_threshold);
}
```

+) Với việc giới hạn ngưỡng sẽ giúp bạn kiểm soát giá trị ADC tốt hơn.

```
cfg.channel = 1; // AIN0
```

```
cfg.pga = 0x0400; // ±2.048V
```

```
cfg.mode = 0x0000; // Chế độ liên tục
```

```
cfg.datarate = 0x0080; // 128 samples/s
```

+) Mẫu cấu hình thiết bị, như các trường hợp đã đưa ra ở mục **5.2.**

```
while (1) {
    if (ioctl(fd, ADS1115_IOCTL_READ_CONFIG, &cfg) < 0) {
        perror("IOCTL read failed");
        break;
    }
}
```

}

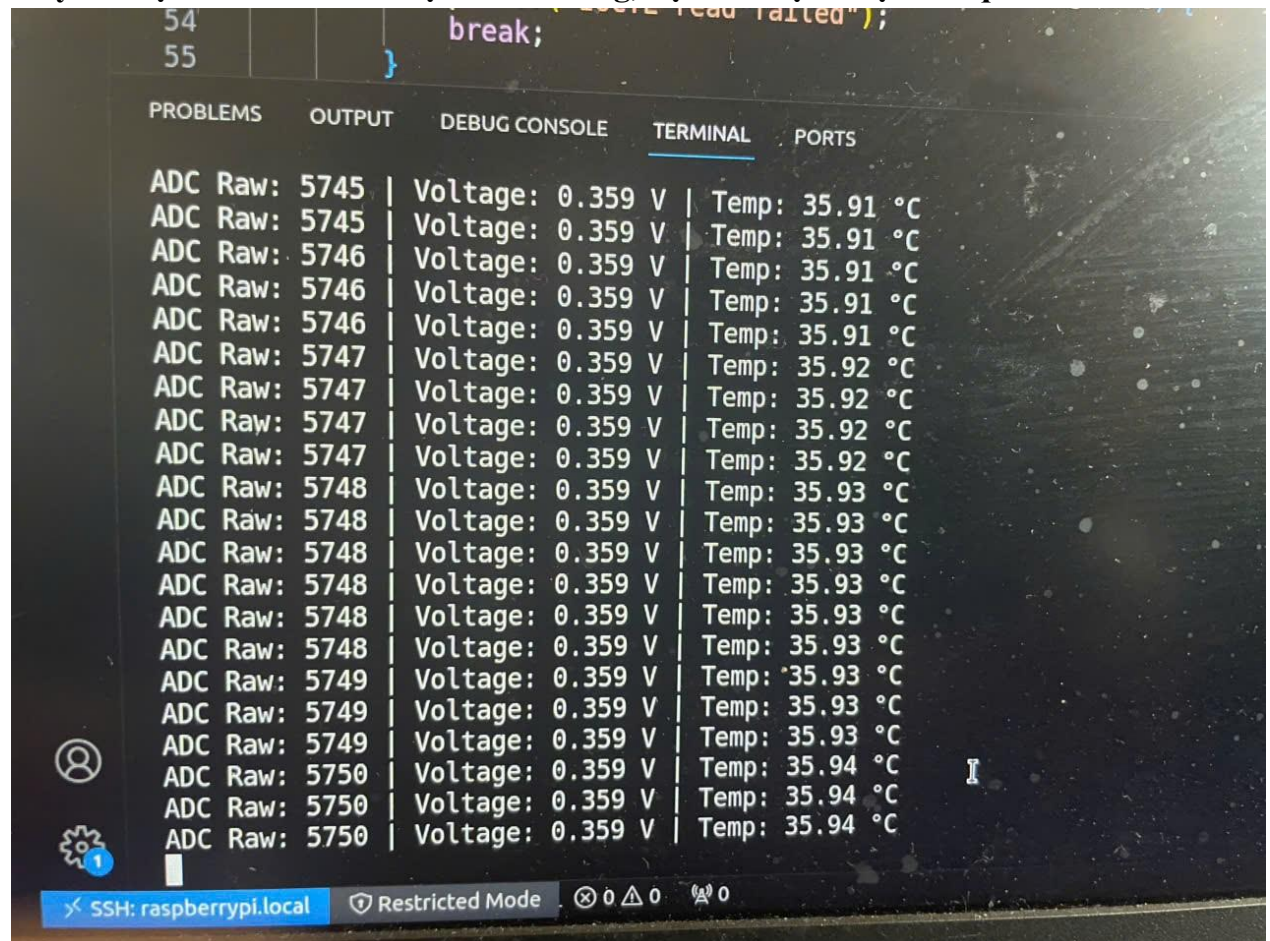
+) Gửi lệnh ioctl để yêu cầu kernel đọc giá trị ADC.

## 6. Tài liệu tham khảo:

Datasheet ADS1115 (TI).

Chúng tôi có để một đoạn code Demo ứng dụng của driver ADS1115, với tín hiệu đầu vào là tín hiệu ADC từ cảm biến nhiệt độ LM35 ở link Github. Nhờ đó bạn có thể hiểu rõ hơn về cách sử dụng của driver.

Hãy cài đặt driver và biên dịch. Nếu đúng, bạn sẽ nhận được kết quả trả về:



The screenshot shows a terminal window with a dark background. At the top, there are tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS'. The 'TERMINAL' tab is selected. The output shows a series of lines for ADC raw values and temperature readings. The raw values range from 5745 to 5750, and the temperature readings range from 35.91 °C to 35.94 °C. The voltage is consistently 0.359 V. At the bottom of the terminal, there is a status bar showing 'SSH: raspberrypi.local', 'Restricted Mode', and some icons.

```
54  
55  
break;  
ADC Raw: 5745 | Voltage: 0.359 V | Temp: 35.91 °C  
ADC Raw: 5745 | Voltage: 0.359 V | Temp: 35.91 °C  
ADC Raw: 5746 | Voltage: 0.359 V | Temp: 35.91 °C  
ADC Raw: 5746 | Voltage: 0.359 V | Temp: 35.91 °C  
ADC Raw: 5746 | Voltage: 0.359 V | Temp: 35.91 °C  
ADC Raw: 5747 | Voltage: 0.359 V | Temp: 35.92 °C  
ADC Raw: 5747 | Voltage: 0.359 V | Temp: 35.92 °C  
ADC Raw: 5747 | Voltage: 0.359 V | Temp: 35.92 °C  
ADC Raw: 5747 | Voltage: 0.359 V | Temp: 35.92 °C  
ADC Raw: 5748 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5748 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5748 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5748 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5748 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5748 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5749 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5749 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5749 | Voltage: 0.359 V | Temp: 35.93 °C  
ADC Raw: 5750 | Voltage: 0.359 V | Temp: 35.94 °C  
ADC Raw: 5750 | Voltage: 0.359 V | Temp: 35.94 °C  
SSH: raspberrypi.local Restricted Mode 0 0 0
```