

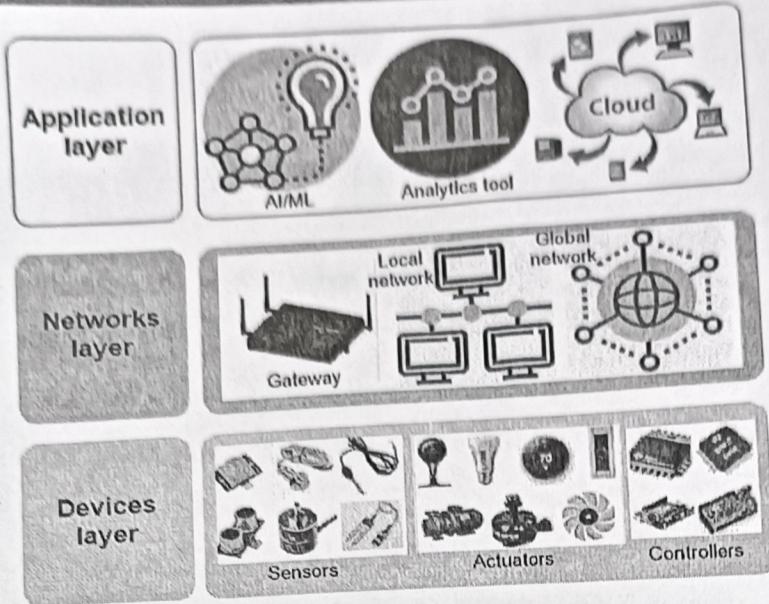
7. Lập trình trao đổi dữ liệu qua giao thức Lora

- Mục đích
 - Lập trình trao đổi dữ liệu thông qua giao thức Lora giữa mô-đun “IoT Gateway” và các mô-đun “IoT Node”.
 - Thiết kế các FRAME truyền đáp ứng yêu cầu trao đổi dữ liệu.
 - Lập trình giao tiếp, thu thập dữ liệu và điều khiển các thiết bị ngoại vi cơ bản với Raspberry Pi.
 - Xây dựng các ứng dụng IoT cơ bản với Raspberry Pi.

- Yêu cầu
 - Xây dựng được FRAME truyền đáp ứng yêu cầu trao đổi dữ liệu giữa mô-đun “IoT Gateway” và các mô-đun “IoT Node”.
 - Giải thích chi tiết ý nghĩa các thông tin trong FRAME truyền.
 - Biết cách lập trình giao tiếp, thu thập dữ liệu và điều khiển các thiết bị ngoại vi cơ bản với Raspberry Pi thông qua giao thức Lora giữa mô-đun “IoT Gateway” và các mô-đun “IoT Node”.

7.1. Kiến trúc IoT

Mô hình kiến trúc IoT đóng vai trò quan trọng để có thể thiết kế một ứng dụng IoT trong thực tiễn. Trong thực tế mô hình kiến trúc IoT được chia thành nhiều lớp khác nhau. Chẳng hạn như, mô hình kiến trúc 5 lớp bao gồm: lớp cảm biến, lớp truy nhập dữ liệu, lớp mạng, lớp middleware và lớp ứng dụng. Một số ứng dụng tiếp cận kiến trúc IoT 4 lớp bao gồm lớp cảm biến và thiết bị chấp hành, lớp thu thập dữ liệu và Internet Gateway, lớp thực hiện điện toán biên, lớp trung tâm dữ liệu điện toán. Tùy vào quy mô hệ thống mà mô hình kiến trúc có thể từ 3 lớp đến nhiều lớp. Trong phạm vi phòng thí nghiệm, mô hình kiến trúc IoT được thiết kế bao gồm 3 lớp cơ bản đó là lớp thiết bị, lớp mạng và lớp ứng dụng. Mô hình kiến trúc này được mô tả như hình minh họa bên dưới.



- Lớp thiết bị:

- Lớp này bao gồm các cảm biến, thiết bị chấp hành và các bộ điều khiển như vi xử lý/vi điều khiển, PLC, FPGA đến các máy tính nhúng.
- Lớp thiết bị thực hiện đo lường và thu thập dữ liệu các đại lượng vật lý thông qua các cảm biến, điều khiển các thiết bị chấp hành và có thể truyền và nhận dữ liệu từ các thiết bị khác qua mạng.

- Lớp mạng:

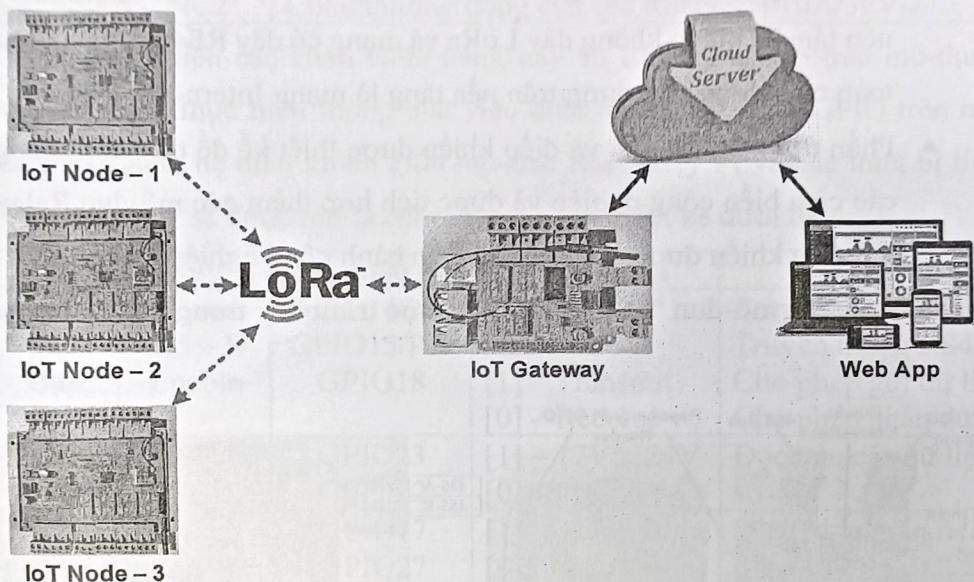
- Chức năng lớp mạng xác định các giao thức truyền thông khác nhau được sử dụng cho việc kết nối mạng và thực hiện điện toán biên.
- Lớp mạng bao gồm các thiết bị liên kết mạng như Hub, Switch, Router; các thiết bị chuyển đổi giao thức mạng như Gateways; đến các thiết bị có khả năng lưu trữ, xử lý cục bộ trước khi gửi dữ liệu lên Server trung tâm.
- Các “Things” ở lớp thiết bị được kết nối với thiết bị Gateway ở lớp mạng thông qua các mạng cục bộ như Wifi, Zigbee, Bluetooth, LoRaWAN ... đến các mạng có dây như CAN, Modbus, Profibus, RS485, Ethernet,... Sau đó, thiết bị ở lớp mạng thực hiện xử lý và

gửi lên trung tâm dữ liệu qua mạng toàn cầu như Internet, 3G/4G/LTE, GSM.

- Lớp ứng dụng:

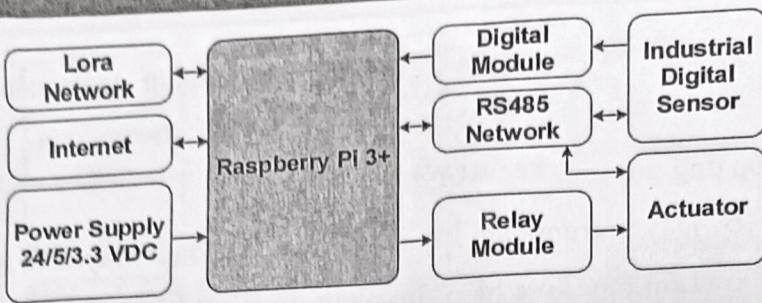
- Đây là trung tâm lưu trữ dữ liệu hay đám mây điện tử.
- Lớp này thực hiện thu nhận dữ liệu từ lớp mạng, lưu trữ, xử lý dữ liệu và ra quyết định dựa trên các thuật toán AI/ML hoặc các công cụ phân tích dữ liệu hiện đại.

Sơ đồ khái niệm các thành phần mô hình thí nghiệm IoT được thiết kế như hình minh họa bên dưới, trong đó bao gồm 3 thành phần chính như sau: “IoT Node”, “IoT Gateway” và máy chủ (Server).



7.2. Cấu trúc mô-đun “IoT Gateway”

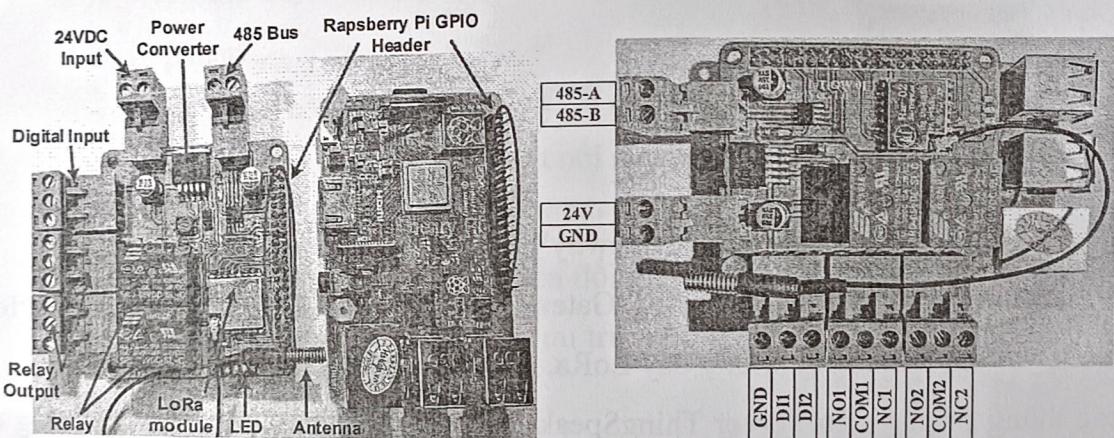
Chức năng của mô-đun “IoT Gateway” là nhận thông tin được gửi từ các “IoT Node” thông qua mạng không dây LoRa, xử lý các thông tin nhận được và sau đó gửi các thông tin này đến Server ThingSpeak. Hơn nữa, mô-đun “IoT Gateway” cũng có khả năng thu thập được các thông tin cảm biến và điều khiển được các thiết bị chấp hành công nghiệp. Sơ đồ khái niệm của mô-đun “IoT Gateway” được trình bày trong hình minh họa bên dưới.



Các thành phần chính của mô-đun bao gồm: Bộ xử lý trung tâm, phần truyền thông, phần thu thập dữ liệu và điều khiển.

- Bộ xử lý trung tâm được xây dựng dựa trên nền tảng là mô-đun Raspberry Pi 4.
- Phần truyền thông bao gồm truyền thông cục bộ (được xây dựng dựa trên nền tảng là mạng không dây LoRa và mạng có dây RS485) và truyền thông toàn cầu (được xây dựng trên nền tảng là mạng Internet).
- Phần thu thập dữ liệu và điều khiển được thiết kế để thu thập thông tin từ các cảm biến công nghiệp và được tích hợp thêm các mô-đun Relay để có thể điều khiển được các thiết bị chấp hành công nghiệp.

Phần cứng của mô-đun “IoT Gateway” được trình bày trong hình minh họa bên dưới.



Công dụng và thông số làm việc của các chân chức năng được tích hợp sẵn trên mô-đun “IoT Gateway” được mô tả trong bảng liệt kê dưới đây.

STT	Tên chân	Công dụng	Thông số
1	24V	Ngõ vào cung cấp điện áp nguồn cho hệ thống hoạt động.	$V_{IN} = 24V_{DC}$ $I_{IN(MIN)} = 2000mA$
2	GND		
3	485-A	Cổng giao tiếp nối tiếp theo giao thức RS485.	
4	485-B		
5	DI1 – DI2	Ngõ vào tín hiệu số.	Logic 1: 17V – 24V Logic 0: 0V – 7V
6	NO1 – NO2	Chân thường hở của các Relay.	Tiếp điểm:
7	COM1 – COM2	Chân chung của các Relay.	$10A/250V_{AC}$
8	NC1 – NC2	Chân thường đóng của các Relay.	$10A/30V_{DC}$

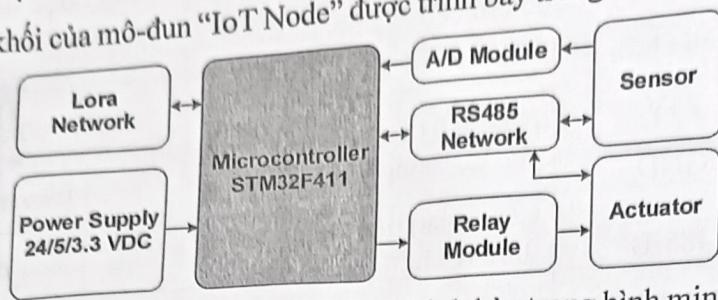
Việc điều khiển các chân chức năng này và LED tích hợp trên mô-đun “IoT Gateway” sẽ được thực hiện thông qua việc điều khiển các chân GPIO trên mô-đun Raspberry Pi. Quan hệ điều khiển giữa mô-đun Raspberry Pi và các thiết bị tích hợp trên “IoT Gateway” sẽ được mô tả chi tiết trong bảng liệt kê dưới đây.

	“IoT Gateway”		Raspberry	Điều khiển	Công dụng
1	485 Bus	485 – A 485 – B Enable	GPIO14/TX GPIO15/RX GPIO18 [1] – Transmit [0] – Receive	Truyền thông RS485 Truyền thông RS485 Cho phép gửi dữ liệu Cho phép nhận dữ liệu
2	Digital Input	DI1 DI2	GPIO23 GPIO22	[1] – 17V...24V [0] – 0V...7V	Đọc vào các dữ liệu số
3	Relay Output	Relay 1 Relay 2	GPIO17 GPIO27	[1] – Relay đóng [0] – Relay ngắt	Điều khiển các relay đóng hoặc ngắt
4	LED	LED 1 LED 2 LED 3 LED 4	GPIO5 GPIO6 GPIO13 GPIO19	[1] – LED bật [0] – LED tắt	Điều khiển các LED đơn bật hoặc tắt

7.3.Cấu trúc mô đun IoT Node

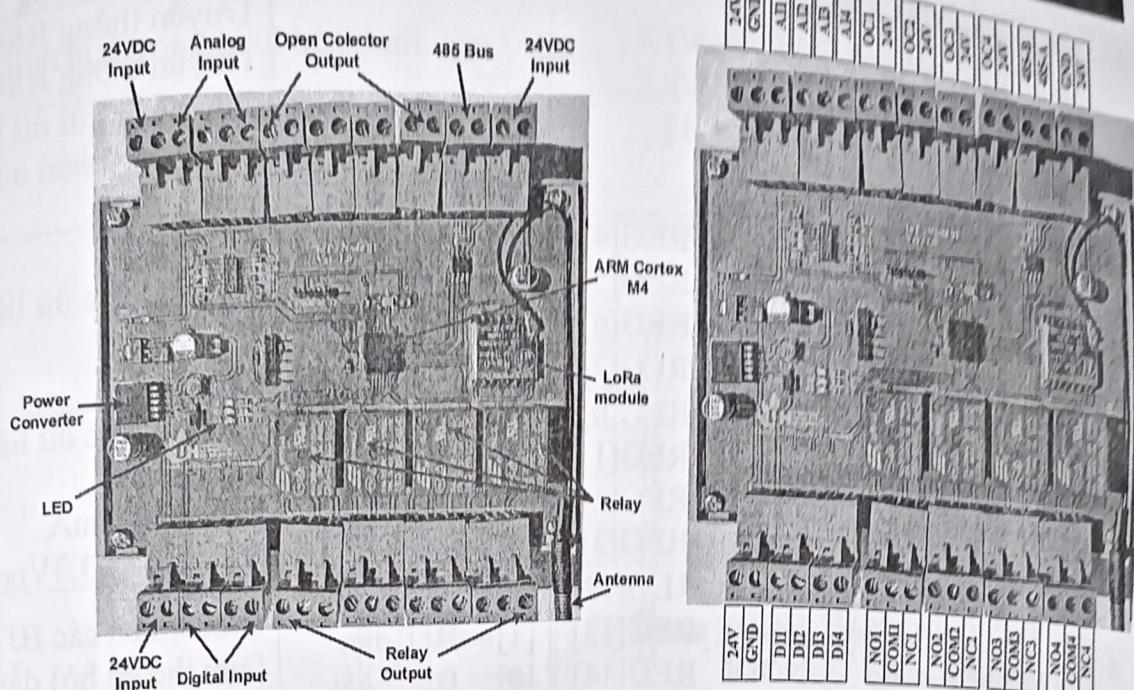
Chức năng của mô-đun “IoT Node” là thu thập thông tin của các cảm biến thông qua mô-đun A/D hoặc thông qua truyền thông nối tiếp (RS232, RS485), song song (SPI, I2C) và xuất tín hiệu điều khiển các thiết bị chấp hành thông qua mô-đun A/D, PWM hoặc truyền thông nối tiếp (RS232, RS485), song song (SPI, I2C). Mô-đun “IoT Node”

sẽ thực hiện việc truyền thông với mô-đun “IoT Gateway” thông qua mạng không dây LoRa. Sơ đồ khối của mô-đun “IoT Node” được trình bày trong hình minh họa bên dưới.



Phần cứng của mô-đun “IoT Node” được trình bày trong hình minh họa bên dưới. Các thành phần chính của mô-đun bao gồm: Bộ xử lý trung tâm, phần truyền thông, phần thu thập dữ liệu và điều khiển, phần nguồn.

- Bộ xử lý trung tâm được xây dựng dựa trên nền tảng là vi điều khiển STM32F411 (ARM Cortex M4) chạy ở tốc độ 100 MHz. Họ vi điều khiển STM32F411 tích hợp 256 – 512KB Flash và 128 KB SRAM, 3 cổng USART chạy với tốc độ lên tới 12,5Mbit/s, 5 cổng SPI (đa hợp với I2S) chạy với tốc độ lên tới 50 Mbit/s, 3 cổng I²C , 1 cổng SDIO chạy ở tốc độ 48 MHz, ADC 12 bit đạt 2,4 MSPS, 11 bộ định thời 16 và 32 bit.
- Phần truyền thông bao gồm truyền thông mạng không dây LoRa và mạng có dây RS485. Phần cứng LoRa được thiết kế dựa trên mô-đun LoRa Ra-02 SX1278 được kết nối với chân SPI của vi điều khiển và phần cứng RS485 được thiết kế dựa trên vi mạch MAX485 được kết nối với các chân TXD và RXD của vi điều khiển.
- Phần thu thập dữ liệu và điều khiển sẽ thực hiện việc thu thập thông tin cảm biến thông qua mô-đun ADC hoặc giao tiếp nối tiếp RS485 và tạo ra các tín hiệu điều khiển truyền tới các cơ cấu chấp hành thông qua các ngõ ra số, PWM hoặc giao tiếp nối tiếp RS485.
- Phần nguồn được thiết kế để hỗ trợ các mức điện áp nguồn theo chuẩn công nghiệp 24VDC, 5VDC và 3,3VDC cung cấp điện áp làm việc cho tất cả các phần của “IoT Node”.



Đặc tính kỹ thuật và thông số làm việc của các chân chức năng được tích hợp sẵn trên mô-đun “IoT Node” được mô tả trong bảng liệt kê bên dưới.

STT	Tên chân	Công dụng	Thông số
1	24V	Ngõ vào cung cấp điện áp nguồn cho hệ thống hoạt động.	$V_{IN} = 24V_{DC}$ $I_{IN(MIN)} = 2000mA$
2	GND		
3	485-A	Cổng giao tiếp nối tiếp theo giao thức RS485.	
4	485-B		
5	DI1 – DI4	Ngõ vào tín hiệu số.	Logic 1: 17V – 24V Logic 0: 0V – 7V
6	AI1 – AI4	Ngõ vào tín hiệu tương tự (ADC có độ phân giải 12bit).	$I_{IN} = 0 – 20mA$ $V_{IN} = 0V – 3,3V_{DC}$
7	OC1 – OC4	Ngõ ra cực thu hở.	$I_{SINK(MAX)} = 100mA$
8	NO1 – NO4	Chân thường hở của các Relay.	Tiếp điểm: 10A/250V _{AC} 10A/30V _{DC}
9	COM1 – COM4	Chân chung của các Relay.	
10	NC1 – NC4	Chân thường đóng của các Relay.	

Việc điều khiển các chân chức năng này sẽ được thực hiện thông qua việc điều khiển các thanh ghi chức năng đặc biệt trong vi điều khiển STM32F411. Quan hệ điều khiển giữa các thanh ghi chức năng đặc biệt trong vi điều khiển STM32F411 và các thiết bị tích hợp trên mô-đun “IoT Node” được mô tả chi tiết ở bảng liệt kê bên dưới.

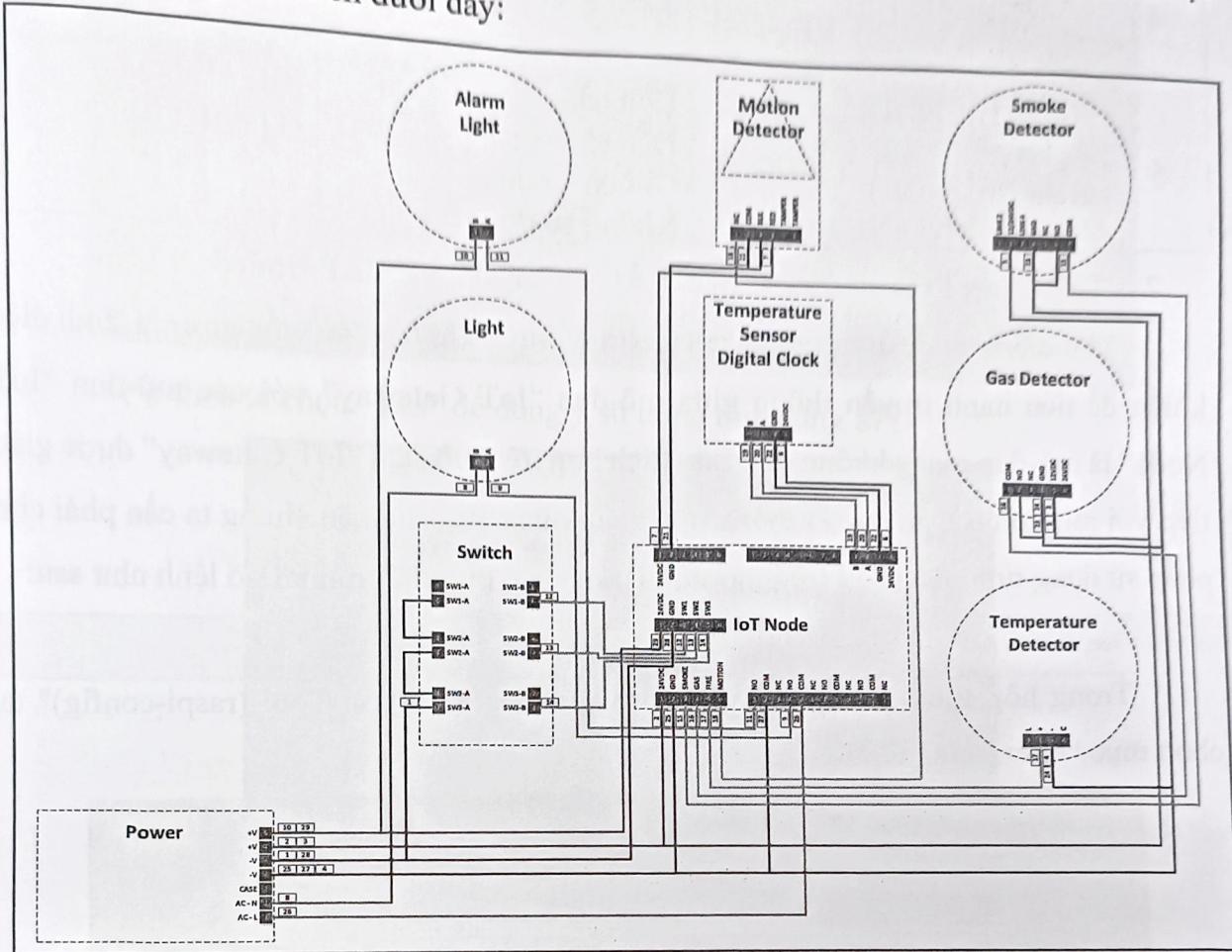
STT	“IoT Node”		STM32F411	Điều khiển	Công dụng
1	485 Bus	485 – A 485 – B Enable	PB6/TX PB7/RX PB7 [1] – Transmit [0] – Receive	Truyền thông RS485 Truyền thông RS485 Cho phép gửi dữ liệu Cho phép nhận dữ liệu
2	Digital Input	DI1 DI2 DI3 DI4	NODE_REG[4] NODE_REG[5] NODE_REG[6] NODE_REG[7]	[1] – 17V...24V [0] – 0V...7V	Đọc vào các dữ liệu số
3	Analog Input	AI1 AI2 AI3 AI4	NODE_REG[0] NODE_REG[1] NODE_REG[2] NODE_REG[3]	Dãy giá trị ADC (ADC có độ phân giải 12bit) [0]...[4096]	Đọc vào các dữ liệu tương tự $I_{IN} = 0 - 20mA$ $V_{IN} = 0V - 3,3V_{DC}$
4	OC Output	OC1 OC2 OC3 OC4	NODE_REG[12] NODE_REG[13] NODE_REG[14] NODE_REG[15]	[1] – BJT dẫn [0] – BJT ngắt	Điều khiển các BJT (cực thu để hở) dẫn hoặc ngắt
5	Relay Output	Relay 1 Relay 2 Relay 3 Relay 4	NODE_REG[8] NODE_REG[9] NODE_REG[10] NODE_REG[11]	[1] – Relay đóng [0] – Relay ngắt	Điều khiển các relay đóng hoặc ngắt

7.4. Sơ đồ kết nối các cảm biến và thiết bị chấp hành trên mô hình IoT

Trên mô hình IoT được thiết kế với các thành phần sau:

- Thiết bị chấp hành:
 - Đèn chiếu sáng (Light).
 - Đèn cảnh báo (Alarm light).
 - Đồng hồ thời gian số (Digital Clock).
- Cảm biến:
 - Phát hiện chuyển động (Motion Detector).
 - Phát hiện khói (Smoke Detector).
 - Phát hiện gas (Gas Detector).
 - Phát hiện quá nhiệt (Temperature Detector).
 - Cảm biến nhiệt độ/độ ẩm (Temperature/Humidity Sensor).

Sơ đồ kết nối tổng quát các cảm biến và thiết bị chấp hành trên mô hình IoT được minh họa trong hình bên dưới đây:



Quan hệ điều khiển của các thiết bị tích hợp trên “IoT Gateway” sẽ được mô tả chi tiết trong bảng liệt kê dưới đây.

STT	Node	STM32F411	Thiết bị	Quan hệ điều khiển
1	485 – A 485 – B	NODE_REG[16] NODE_REG[17] NODE_REG[18] NODE_REG[19] NODE_REG[20]	Nhiệt độ LM35A Nhiệt độ LM35B Nhiệt độ DHT22 Độ ẩm DHT22 Đồng hồ	Chỉ đọc giá trị 359 → 35,9°C 563 → 56,3% 1523 → 15g23
2	DI1 DI2 DI3 DI4	NODE_REG[4] NODE_REG[5] NODE_REG[6] NODE_REG[7]	SW1 hoặc Smoke SW2 hoặc Gas SW3 hoặc Temperature Motion	[1] – SW tắt / Cảm biến bình thường [0] – SW bật / Cảm biến cảnh báo
3	AI1 AI2 AI3 AI4	NODE_REG[0] NODE_REG[1] NODE_REG[2] NODE_REG[3]	Không sử dụng	

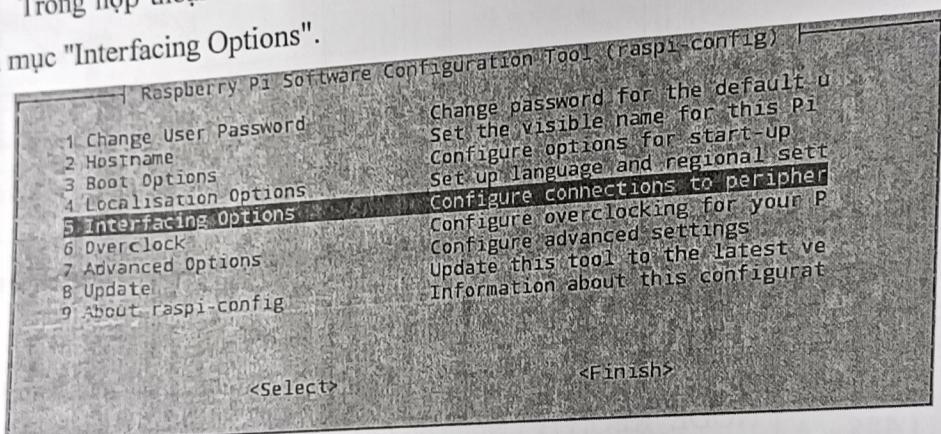
	OC1	NODE_REG[12]	Không sử dụng	
	OC2	NODE_REG[13]		
	OC3	NODE_REG[14]		
	OC4	NODE_REG[15]		
5	Relay 1	NODE_REG[8]	Đèn báo động	[0] – Tắt đèn [1] – Bật đèn
	Relay 2	NODE_REG[9]	Đèn chiếu sáng	
	Relay 3	NODE_REG[10]	Không sử dụng	
	Relay 4	NODE_REG[11]	Không sử dụng	

7.5. Lập trình Python điều khiển "IoT Gateway" và "IoT Node"

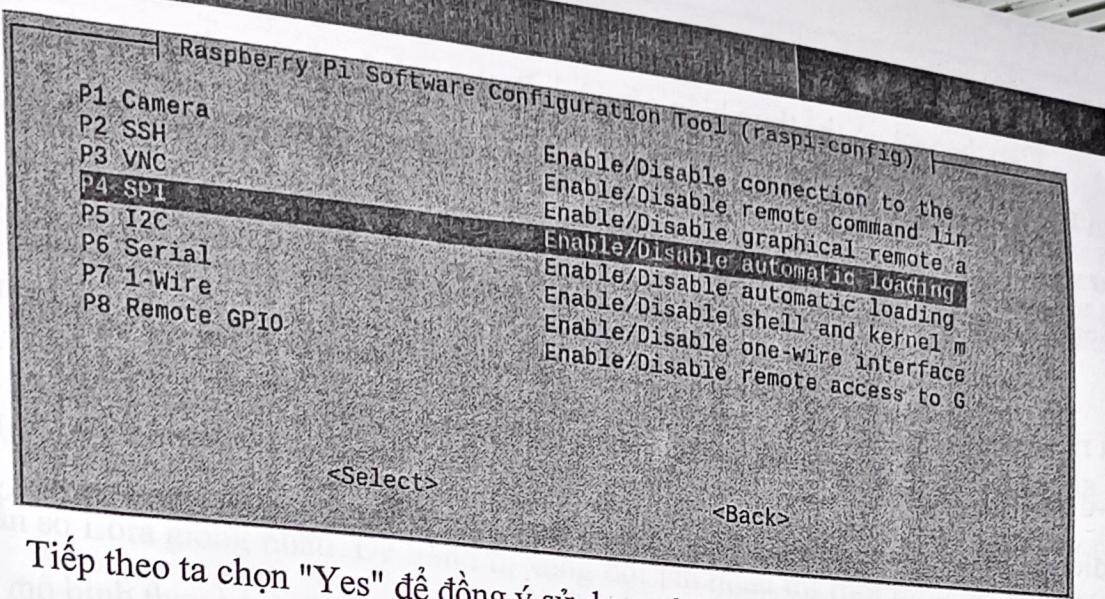
Một điểm rất quan trọng mà ta cần lưu ý trước khi bắt đầu viết chương trình điều khiển để tiến hành truyền thông giữa mô-đun "IoT Gateway" với các mô-đun "IoT Node" là mô-đun mạng không dây Lora tích hợp trên mô-đun "IoT Gateway" được giao tiếp với mô-đun Raspberry Pi thông qua giao thức SPI. Cho nên chúng ta cần phải cho phép sử dụng tính năng SPI trên mô-đun Raspberry Pi, tại Terminal gõ lệnh như sau:

```
~$ sudo raspi-config
```

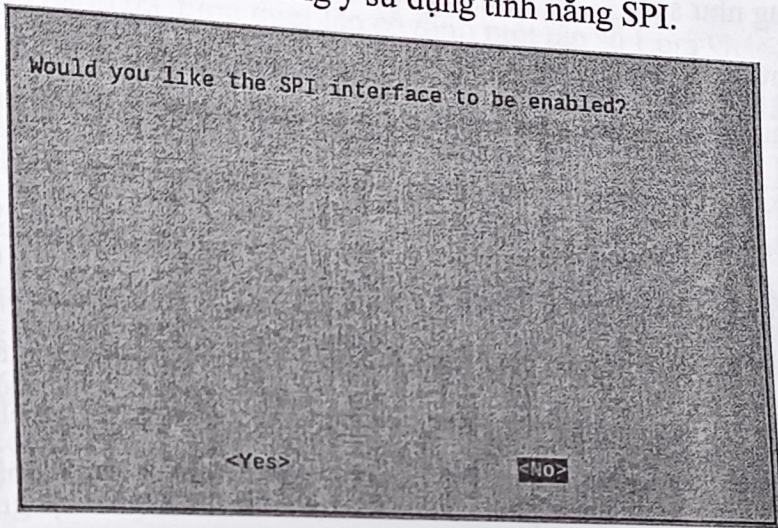
Trong hộp thoại "Raspberry Pi Software Configuration Tool (raspi-config)" ta chọn mục "Interfacing Options".



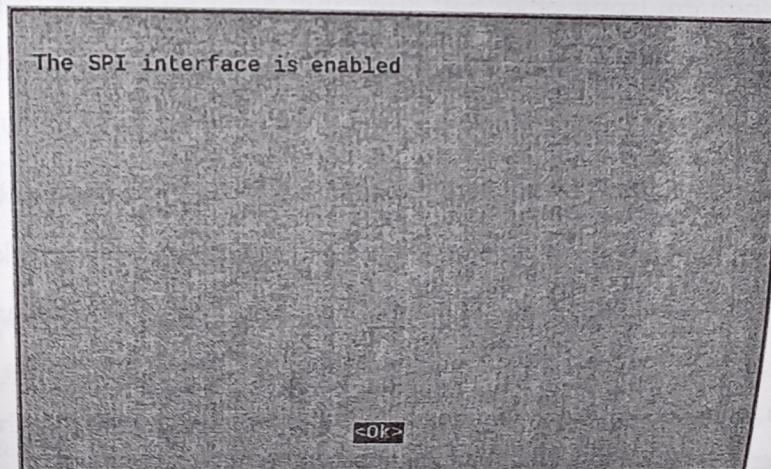
Tiếp theo ta chọn vào mục "SPI" để thiết lập sử dụng tính năng SPI.



Tiếp theo ta chọn "Yes" để đồng ý sử dụng tính năng SPI.



Cuối cùng hoàn tất việc thiết lập bằng cách chọn "OK".



Tiếp theo để có thể thực hiện các lệnh Python sử dụng cho giao tiếp SPI trên mô-đun Raspberry Pi thì ta cần cài đặt thêm gói thư viện “spidev”, tại Terminal nhập lệnh như sau:

```
~$ sudo pip3 install spidev
```

Trong phần này chúng ta sẽ thực hiện một số bài thí nghiệm đơn giản để cho thấy khả năng truyền thông qua mạng không dây Lora giữa mô-đun “IoT Gateway” và các mô-đun “IoT Node” khác nhau, nhằm thực hiện việc thu thập dữ liệu từ các cảm biến và điều khiển các cơ cấu chấp hành. Các bài thí nghiệm trong phần này sẽ được chia ra làm hai nội dung như sau:

- Thứ nhất sẽ tập trung vào việc lập trình điều khiển trên mô-đun “IoT Gateway” để thực hiện việc thu thập dữ liệu số (Digital Input), điều khiển các cơ cấu chấp hành (Relay Output) và truyền thông có dây qua giao thức RS485.
- Thứ hai sẽ tập trung vào việc lập trình điều khiển trên các mô-đun “IoT Node” khác nhau để thu thập dữ liệu số (Digital Input), dữ liệu tương tự (Analog Input), điều khiển các cơ cấu chấp hành (Relay Output, OC Output) và truyền thông có dây qua giao thức RS485 hoặc truyền thông không dây qua mạng Lora.

Các bài thí nghiệm trong phần này được viết bằng ngôn ngữ Python và sử dụng mạng truyền thông không dây Lora trong việc truyền nhận dữ liệu giữa mô-đun “IoT Gateway” và các mô-đun “IoT Node” khác nhau. Tuy nhiên thư viện dành cho mạng truyền thông không dây Lora lại không được cài đặt sẵn trong mô-đun Raspberry Pi. Vì vậy, để có thể sử dụng được bộ thư viện dành cho mạng truyền thông không dây Lora thì việc đầu tiên là phải tiến hành cài đặt bộ thư viện này vào mô-đun Raspberry Pi chạy hệ điều hành Raspbian. Việc cài đặt bộ thư viện này rất đơn giản, chúng ta chỉ việc sao chép nó vào trong cùng thư mục với tập tin mã nguồn của chương trình điều khiển và bổ sung các dòng lệnh khai báo sau đây vào mã nguồn:

```
import sys  
sys.path.append('/home/pi/Desktop/Demo/Lora_Driver/')  
from Lora_Driver.IoT_Driver import myLora
```

```
Lora = myLora(verbose = False)
```

Trong các dòng khai báo trên ta cần phải chú ý là tùy theo việc lưu trữ thư mục “Lora_Driver” ở đâu mà phải hiệu chỉnh và khai báo chính xác đường dẫn của thư mục này tại dòng lệnh minh họa dưới đây.

```
sys.path.append('/home/pi/Desktop/Demo/Lora_Driver/')
```

Tiếp theo để có thể thực hiện trong truyền thông bằng giao thức Lora giữa mô-đun “IoT Gateway” và mô-đun “IoT Node” thì đòi hỏi cả hai mô-đun này phải có giá trị tần số Lora giống nhau. Để tránh bị xung đột lẫn nhau thì mỗi mô-đun “IoT Node” trên mô hình thực hành đã được thiết lập cố định một tần số Lora khác nhau, giá trị tần số Lora của từng mô hình được mô tả chi tiết trong bảng liệt kê dưới đây.

Mô hình	Tần số Lora	Mô hình	Tần số Lora
1	439 MHz	7	469 MHz
2	444 MHz	8	474 MHz
3	449 MHz	9	479 MHz
4	454 MHz	10	484 MHz
5	459 MHz	11	489 MHz
6	464 MHz	12	494 MHz

Tần số Lora của mô-đun “IoT Gateway” trên mô hình thực hành có thể được điều chỉnh bằng dòng lệnh Python như sau.

```
Lora.set_freq(freq_val)
```

Để thực hiện việc ghi dữ liệu từ mô-đun “IoT Gateway” đến mô-đun “IoT Node” cụ thể, được thực hiện bằng dòng lệnh Python như sau.

```
Lora.write_data(node, reg, value)
```

Để thực hiện việc đọc dữ liệu từ mô-đun “IoT Node” cụ thể về mô-đun “IoT Gateway”, được thực hiện bằng dòng lệnh Python như sau.

```
value_read = Lora.read_data(node, reg)
```

Trong đó:

- freq_val: Là giá trị tần số Lora (đơn vị MHz).

- **node:** Là địa chỉ của mô-đun “IoT Node”. Địa chỉ này đã được mặc định bởi nhà sản xuất thiết bị, chúng ta cần liên hệ với nhà sản xuất để biết giá trị địa chỉ cụ thể cho từng Node.
- **reg:** Là địa chỉ của thanh ghi trong Node mà ta cần ghi (hoặc đọc) dữ liệu tương ứng cho từng thiết bị được trang bị trên Node. Địa chỉ này đã được mặc định bởi nhà sản xuất thiết bị, chúng ta cần tham khảo trong bảng đã trình bày bên trên để biết giá trị địa chỉ cụ thể cho từng thiết bị.
- **value:** Là giá trị cần ghi vào thanh ghi.
- **value_read:** Là biến chứa dữ liệu đọc được từ thanh ghi của Node.

Đoạn chương trình dưới đây minh họa chi tiết cho việc mô-đun “IoT Gateway” liên tục đọc trạng thái hiện tại của công tắc 1 (Switch) để điều khiển trạng thái bật/tắt của đèn chiếu sáng (Light), ví dụ được sử dụng cho “Mô hình IoT System - 1” như sau.

```
#!/usr/bin/env python3
import sys
sys.path.append('/home/pi/Desktop/Code_V5/Lab_9_Lora/Demo/Lora_Driver')
from Lora_Driver.IoT_Driver import mylora
Lora = mylora(verbose=False)
Lora.debug_on = 0 # [1]-Debug on; [0]-Debug off
from time import sleep
Lora.set_freq(439)
while(1):
    val_sw1 = Lora.read_data(1,4) # Read SW 1
    print(val_sw1)
    Lora.write_data(1,9,val_sw1) # Control Light
    sleep(0.1)
```

7.6. Báo cáo thực hành

Báo cáo được thực hiện theo nhóm thực hành và nộp về cho giáo viên dưới dạng một tập tin Word theo yêu cầu:

- Nội dung thực hiện báo cáo thì sinh viên xem trong "*Tài liệu báo cáo thực hành môn Internet vạn vật*".
- Tải tập tin báo cáo lên trang web E-Learning (OCW FET-IUH) của Khoa Công nghệ Điện tử.
- Gửi tập tin báo cáo về địa chỉ mail của giáo viên hướng dẫn.

- Tiêu đề mail có dạng: IOT_x_Nhom_n_Bai_m

Trong đó: x là ký hiệu buổi học (CT2, ST5, CT5 hoặc CT6).
 n là số nhóm thực hành (1, 2, 3,...).
 m = 9: Bài “Lập trình trao đổi dữ liệu qua Lora”.

7.7. Tóm tắt nội dung bài học

Nội dung chính của bài học bao gồm:

- Kiến trúc IoT.
- Cấu trúc của mô-đun “IoT Gateway”.
- Cấu trúc của mô-đun “IoT Node”.
- Sơ đồ kết nối các cảm biến và thiết bị chấp hành trên mô hình IoT.
- Lập trình Python điều khiển “IoT Gateway” và “IoT Node”.