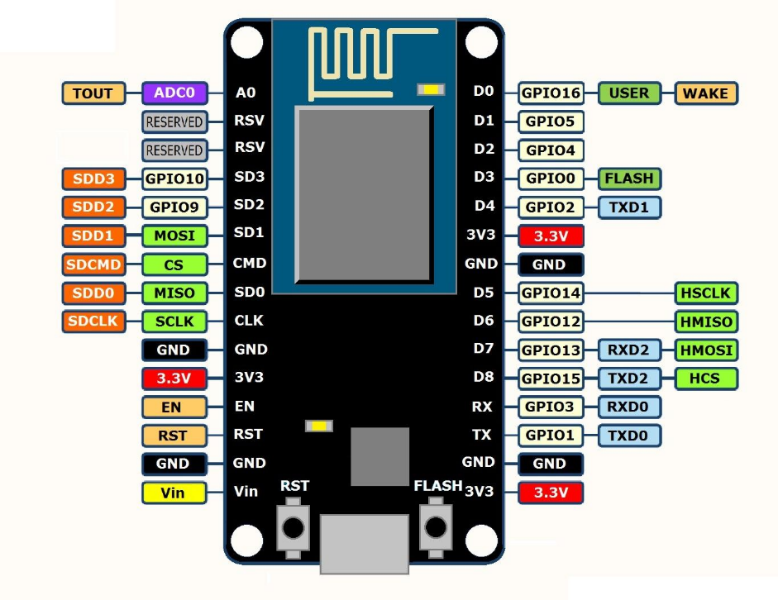
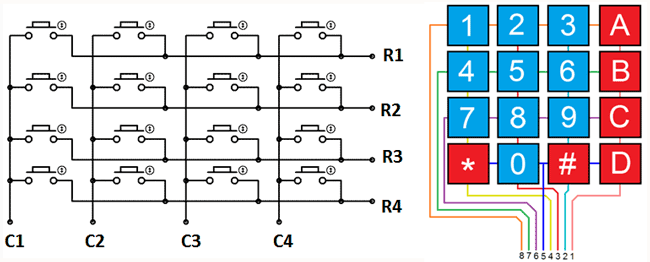
**BÁO CÁO TUẦN 1**

**PHẦN I: VỀ PHẦN CỨNG**

1. **NodeMCU:**

* **Giới thiệu về NodeMCU**
  + NodeMCU là kit phát triển trên các version của ESP8266 với 13 chân GPIO, 1 chân giao tiếp ADC, 1 bộ giao tiếp UART, 1 bộ giao tiếp I2C và hỗ trợ PWM.
* **Tích hợp trên Kit:** 
  + 2 nút nhấn (RST và FLASH)
  + Led hiển thị
  + Chip chuyển USB – UART CP2102
* **Thông số kỹ thuật:**
  + Wifi: hỗ trợ chuẩn 802.11 b/g/n
  + Điện áp hoạt động: 3.3V
  + Điện áp vào: thông qua USB hoặc chân Vin là 5V
  + Bộ nhớ Flash: 4MB
  + Hỗ trợ bảo mật: WPA/WPA2
  + Tích hợp giao thức TCP/IP
  + Khởi tạo và truyền gói tin với tốc độ <2ms
  + Dòng tiêu thụ ở Standby Mode < 1.0mW (DTIM3)

1. **Keypad:**

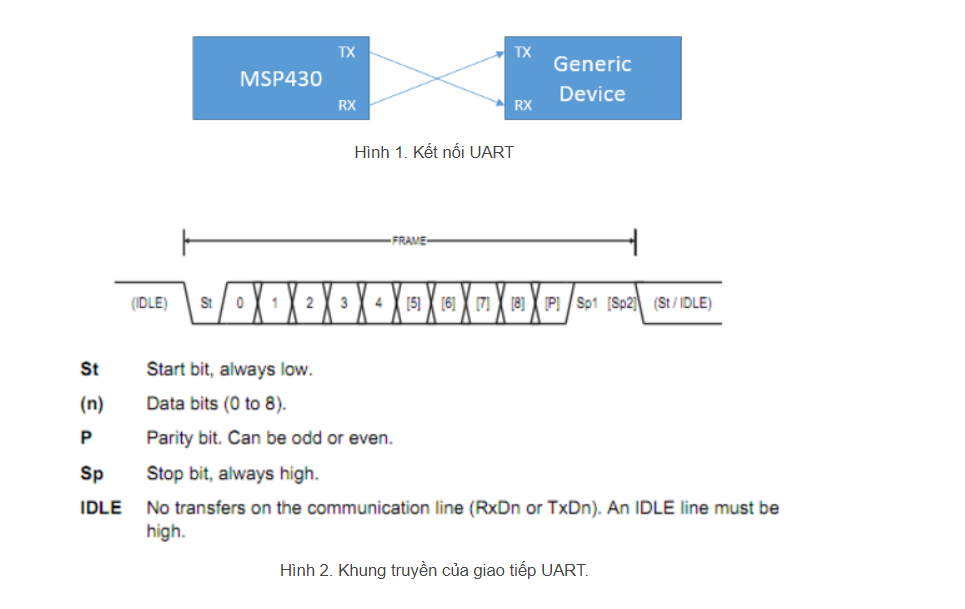
* **Giới thiệu về KeyPad**
  + Keypad là một thiết bị nhập chứa các nút bấm cho phép người dùng nhập các chữ số, chữ cái hoặc ký tự điều khiển. KeyPad không chứa tất cả bảng mã ASCII như keyboard vì thế nó thường được sử dụng trong các ứng dụng chuyên dụng và tương đối đơn giản, ở đó, số lượng nút nhấn thay đổi phụ thuộc vào ứng dụng.
  + KeyPad 4×4 là bàn phím gồm 16 nút nhấn, được xếp thành 4 hàng, mỗi hàng gồm 4 phím bấm như hình dưới đây.
  + Điện áp tối đa trên mỗi nút là: 24V
  + Dòng điện tối đa thông qua nút: 30mA
  + Có 8 chân kết nối (4 row và 4 colum)
* **Nguyên lý hoạt động**
  + Để làm việc với KeyPad 4×4, người lập trình thường sử dụng giải thuật “quét phím”. Giải thuật này yêu cầu vi điêu khiển liên tục đưa các tín hiệu đầu ra ở hàng (hoặc cột) và thu lại đầu vào ở cột (hoặc hàng), nếu phím được bấm, đầu phát tín hiệu sẽ được kết nối với đầu thu, từ đó xác định được phím đã bấm.
  + Việc lựa chọn đầu ra/vào hình thành 2 phương pháp quét phím: theo chiều dọc và theo chiều ngang. Trong báo cáo này, tín hiệu xuất ra ở các hàng và thu lại ở các cột.
  + Giả sử một nút ‘2’ được nhấn, khi đó đường C và 2 được nối với nhau. Nếu đường C được nối với GND, khi đó, điện áp ở chân số 2 sẽ mang điện áp 0V. Tương tự như thế với các phím cùng hàng C.
* **Ví dụ:**
  + Cấp cực âm cho Các chân R1 – R2 – R3 – R4
  + Khi nhấn nút thì các chân bất kì từ thì từ chân C[i] ta sẽ nhận được tín hiệu LOW (Chân nối với C[i] sử dụng INPUT\_PULLUP)
  + Nếu không nhấn nút thì thì các chân C[i] sẽ nhận được tín hiệu HIGH.

**PHẦN II: VỀ PHẦN MỀM.**

1. **GPIO: (General-Purpose Input/Output)**

* GPIO (Cổng đầu vào và ra với mục đích cơ bản) thực tế nó là các chân đầu ra, đầu vào.
* Chúng ta có thể thiết đặt chân là INPUT hay OUTPUT.
* VD: GPIO trong IDE ARDUINO :
  + pinMode(pin\_number, status); // thiết đặt cấu hình chân I/O
  + Nếu status là OUTPUT thì:
    - digitalWrite(pin\_number, values); // thiết values là HIGH hay LOW
  + Nếu status là INPUT thì:
    - digitalRead(pin\_number); // kiểm tra pin\_number đang ở trạng thái HIGH hay LOW

1. **SERIAL MONITOR (UART): Universal Asynchronous Receiver Transmitter:**

* UART là chuẩn giao tiếp truyền nhận dữ liệu không đồng bộ. Đây là chuẩn giao tiếp phổ biến và dễ sử dụng.
* Cách hoạt động: 2 thiết bị muốn giao tiếp UART với nhau thì phải thông qua 2 đường dẫn RX (nhận) và TX (truyền). Vì là không đồng bộ nên 2 thiết bị cần được phải cài đặt thống nhất về khung truyền, tốc độ truyền (tốc độ Baud). (như hình bên dưới)

Start bit: báo hiệu quá trình truyền dữ liệu.

Data bits: dữ liệu cần truyền, thường là 8 bit.

Parity bit: bit kiểm tra chẵn lẻ, dùng để phát hiện lỗi.

Stop bit: báo hiệu kết thúc 1 frame dữ liệu.

* **Về phần Serial Monitor:** 
  + NodeMCU giao tiếp với máy tính thông qua USB to UART. Tức có nghia là sẽ thông qua RX và TX của NodeMCU.
  + Sử dụng các hàm:
    - Serial.begin(9600) : dùng để bật giao tiếp UART với tốc độ Baud là 9600, ngoài ra còn nhiều tốc độ khác có thể thay đổi.
    - Serial.print(“ABC”) :dùng để in ra màn hình Serial Monitor trong IDE Arduino.
    - Serial.write(“ABC”) :dùng để truyền dữ liệu đi.
    - Serial.read() : dùng để đọc dữ liệu được truyền đến.