

Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.

if(millis() > asyncDelay + delayLength){

if(asyncDelay > (4294967295-delayLength)){

asyncDelay = (4294967295-asyncDelay)+(delayLength-(4294967295-asyncDelay));

}else{

asyncDelay+=delayLength;

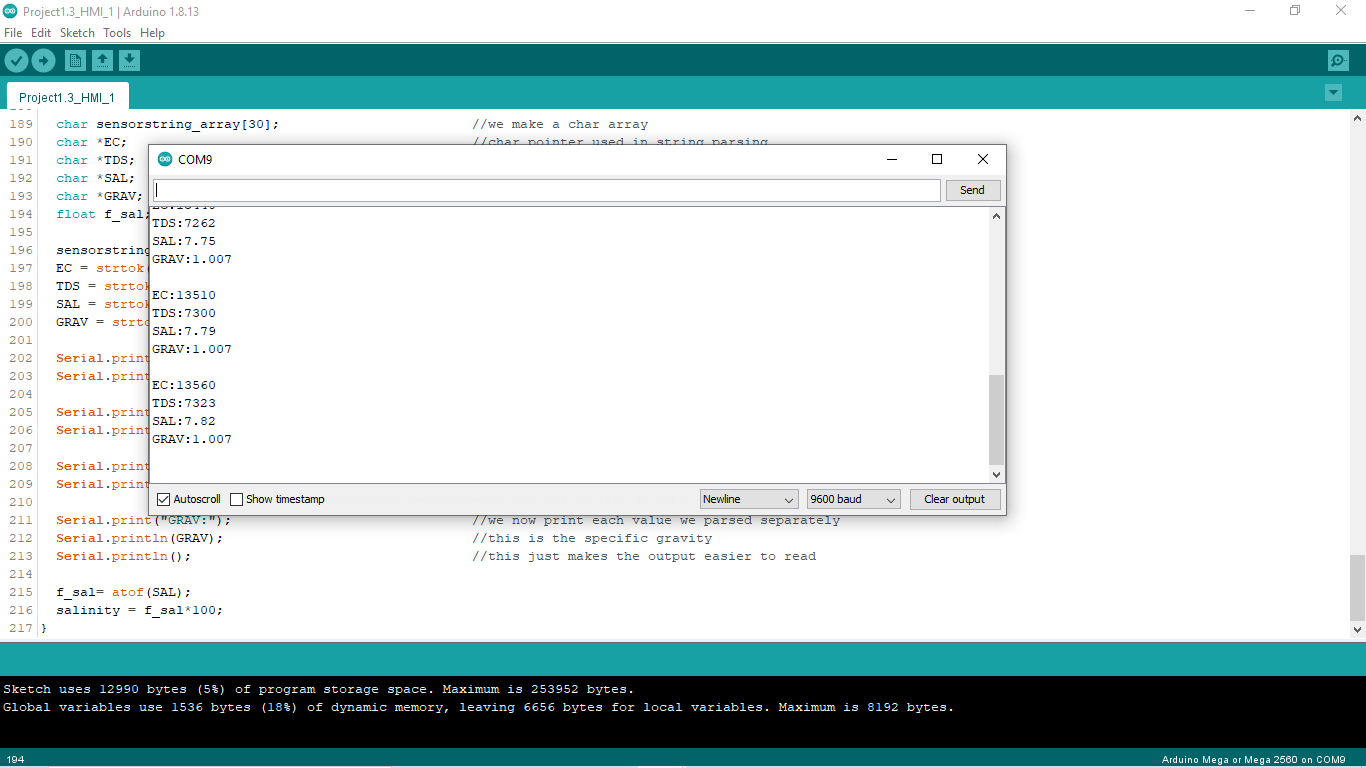
}

}

EC là chữ viết tắt của Electrical Conductivity, hay còn gọi là độ dẫn điện.

  TDS là từ viết tắt của Total Dissolved Solids – Tổng chất rắn hòa tan.

Chỉ số **SG** tên đầy đủ **là** Specific Gravity - tức tỷ trọng. Tỷ trọng nước tiểu được định nghĩa **là** lượng các chất hòa tan hòa tan trong nước tiểu so với nước (=1.000).



#include "Nextion.h"

int CurrentPage = 0;

int temp = 739;

int salinity;

String sensorstring = "";

String inputstring = "";

boolean sensor\_string\_complete = false;

boolean input\_string\_complete = false;

NexButton b20 = NexButton(2, 10, "b20");

NexButton b21 = NexButton(2, 11, "b21");

NexVariable va = NexVariable(2, 8, "va");

NexButton b30 = NexButton(3, 12, "b30");

NexButton b31 = NexButton(3, 14, "b31");

NexButton b32 = NexButton(3, 13, "b32");

NexVariable va0 = NexVariable(3, 10, "va0");

NexPage logo = NexPage(0, 0, "logo");

NexPage disp = NexPage(1, 0, "disp");

NexPage set = NexPage(2, 0, "set");

NexPage calib = NexPage(3, 0, "calib");

char buffer[100] = {0};

NexTouch \*nex\_listen\_list[] =

{

&b20,

&b21,

&b30,

&b31,

&b32,

&va,

&va0,

&logo,

&disp,

&set,

&calib,

NULL

};

void b20PopCallback(void \*ptr) //release bt4

{

memset(buffer, 0, sizeof(buffer));

va.getText(buffer, sizeof(buffer));

Serial.print(buffer);

delay(200);

}

void b21PopCallback(void \*ptr) //release bt5

{

memset(buffer, 0, sizeof(buffer));

va.getText(buffer, sizeof(buffer));

Serial.print(buffer);

delay(200);

}

void b30PopCallback(void \*ptr) //release bt5

{

memset(buffer, 0, sizeof(buffer));

va0.getText(buffer, sizeof(buffer));

Serial.print(buffer);

delay(200);

}

void b31PopCallback(void \*ptr) //release bt5

{

memset(buffer, 0, sizeof(buffer));

va0.getText(buffer, sizeof(buffer));

Serial.print(buffer);

delay(200);

}

void b32PopCallback(void \*ptr) //release bt5

{

memset(buffer, 0, sizeof(buffer));

va0.getText(buffer, sizeof(buffer));

Serial.print(buffer);

delay(200);

}

void logoPushCallback(void \*ptr) // If page 0 is loaded on the display, the following is going to execute:

{

CurrentPage = 0;

Serial.println(CurrentPage);

}

void dispPushCallback(void \*ptr) // If page 0 is loaded on the display, the following is going to execute:

{

CurrentPage = 1;

Serial.println(CurrentPage);

}

void setPushCallback(void \*ptr) // If page 0 is loaded on the display, the following is going to execute:

{

CurrentPage = 2;

Serial.println(CurrentPage);

}

void calibPushCallback(void \*ptr) // If page 0 is loaded on the display, the following is going to execute:

{

CurrentPage = 3;

Serial.println(CurrentPage);

}

void setup() {

Serial1.begin(9600);

Serial.begin(9600);

Serial2.begin(9600);

inputstring.reserve(10);

sensorstring.reserve(30);

b20.attachPop(b20PopCallback);

b21.attachPop(b21PopCallback);

b30.attachPop(b30PopCallback);

b31.attachPop(b31PopCallback);

b32.attachPop(b32PopCallback);

logo.attachPush(logoPushCallback);

disp.attachPush(dispPushCallback);

set.attachPush(setPushCallback);

calib.attachPush(calibPushCallback);

Serial2.print("C,5");

Serial2.print('\r');

}

void serialEvent() { //if the hardware serial port\_0 receives a char

inputstring = Serial.readStringUntil(13); //read the string until we see a <CR>

input\_string\_complete = true; //set the flag used to tell if we have received a completed string from the PC

}

void loop() { //here we go...

delay(30);

Serial1.print("temp.val=");

Serial1.print(temp);

Serial1.write(0xff);

Serial1.write(0xff);

Serial1.write(0xff);

Serial1.print("sal.val=");

Serial1.print(salinity);

Serial1.write(0xff);

Serial1.write(0xff);

Serial1.write(0xff);

nexLoop(nex\_listen\_list);

if (input\_string\_complete == true) { //if a string from the PC has been received in its entirety

Serial2.print(inputstring); //send that string to the Atlas Scientific product

Serial2.print('\r'); //add a <CR> to the end of the string

inputstring = ""; //clear the string

input\_string\_complete = false; //reset the flag used to tell if we have received a completed string from the PC

}

if (Serial2.available() > 0) { //if we see that the Atlas Scientific product has sent a character

char inchar = (char)Serial2.read(); //get the char we just received

sensorstring += inchar; //add the char to the var called sensorstring

if (inchar == '\r') { //if the incoming character is a <CR>

sensor\_string\_complete = true; //set the flag

}

}

if (sensor\_string\_complete == true) { //if a string from the Atlas Scientific product has been received in its entirety

if (isdigit(sensorstring[0]) == false) { //if the first character in the string is a digit

Serial.println(sensorstring); //send that string to the PC's serial monitor

}

else //if the first character in the string is NOT a digit

{

print\_EC\_data(); //then call this function

}

sensorstring = ""; //clear the string

sensor\_string\_complete = false; //reset the flag used to tell if we have received a completed string from the Atlas Scientific product

}

}

void print\_EC\_data(void) { //this function will pars the string

char sensorstring\_array[30]; //we make a char array

char \*EC; //char pointer used in string parsing

char \*TDS; //char pointer used in string parsing

char \*SAL; //char pointer used in string parsing

char \*GRAV; //char pointer used in string parsing

float f\_sal; //used to hold a floating point number that is the EC

sensorstring.toCharArray(sensorstring\_array, 30); //convert the string to a char array

EC = strtok(sensorstring\_array, ","); //let's pars the array at each comma

TDS = strtok(NULL, ","); //let's pars the array at each comma

SAL = strtok(NULL, ","); //let's pars the array at each comma

GRAV = strtok(NULL, ","); //let's pars the array at each comma

Serial.print("EC:"); //we now print each value we parsed separately

Serial.println(EC); //this is the EC value

Serial.print("TDS:"); //we now print each value we parsed separately

Serial.println(TDS); //this is the TDS value

Serial.print("SAL:"); //we now print each value we parsed separately

Serial.println(SAL); //this is the salinity value

Serial.print("GRAV:"); //we now print each value we parsed separately

Serial.println(GRAV); //this is the specific gravity

Serial.println(); //this just makes the output easier to read

f\_sal= atof(SAL);

salinity = f\_sal\*100;

}