Arduino generated program   Sept 9, 2016 Good.

Arduino Ver. 1.8.5

Target hardware ESP8266 micro controller

```
/* Sensus Water Meter Reader
 *  Clk on D5
 *  Data on D6
 *  Relay Set on D1
 *  Relay ReSet on D2
 *
 *  Set Relay
 *
 *  Clock Frequency of 1Khz
 *  Meter Energize time of 600mS
 *
 *  Use EspSoftSerialRX.h to read Serial Data
 *  Print Data
 *
 *  ReSet Relay
 *
 *
 */

// #include <SoftwareSerial.h>
// SoftwareSerial mySerial(D6,D5); //Rx,Tx

#define RxData D6 //Pin 12
#define TxClock D5 //D5(pin 14)
#define led 2
#define SetRelay  5
#define ResetRelay  4

// static const uint8_t D0   = 16;
// static const uint8_t D1   = 5;
// static const uint8_t D2   = 4;
// static const uint8_t D3   = 0;
// static const uint8_t D4   = 2; Led
// static const uint8_t D5   = 14;
// static const uint8_t D6   = 12;
// static const uint8_t D7   = 13;
// static const uint8_t D8   = 15;
// static const uint8_t D9   = 3;
// static const uint8_t D10  = 1;

int cycle = 0;
byte val = 0;
byte clkState = LOW;

unsigned int ab;
unsigned int meterByte[100];

unsigned long count = 0;
unsigned long test = 0;
unsigned long currentMillis = millis();
unsigned long currentMicros = micros();
unsigned long previousMillis = 0;        // For Delay between meter reads
unsigned long previousClkMicros = 0;        // For TxClock timing
unsigned long interval = 3600000;      // 30 sec Time delay between Meter Reads
unsigned long ClkTime = 550;      //[550 100%], [500 No Good], [800 NO], [450 NO], [600 OK]
```

```
}

void setup() {   // put your setup code here, to run once:
  pinMode (RxData,INPUT);      //  Meter Read Pin
  pinMode (TxClock,OUTPUT);                // clock Pin
  pinMode (led,OUTPUT);
  pinMode (SetRelay,OUTPUT);
  pinMode (ResetRelay,OUTPUT);

   Serial.begin(115200);//115200
 digitalWrite(led,HIGH);// off
 digitalWrite(TxClock,LOW); // off

}

void loop() {           // put your main code here, to run repeatedly:
     currentMillis = millis();
  if (currentMillis - previousMillis > interval) {
    previousMillis = currentMillis;
    int x = millis();

    Connect();  // Connect to water meter
   GetData();   // verify data
   DisConnect(); // Disconnect WaterMeter

   int y = millis();
   int z = y-x;
   Serial.print("  Disconnected. Connect time  ");
 Serial.print(z);
  Serial.print("  ");

    DataPrint();



  }
}
```

```
void Connect()
{
  digitalWrite(SetRelay,HIGH);
  delay(10);
  digitalWrite(SetRelay,LOW);

  digitalWrite(led,LOW); // led pin 2
}



void DisConnect()

{
  digitalWrite(ResetRelay,HIGH);
  delay(10);
  digitalWrite(ResetRelay,LOW);

  digitalWrite(led,HIGH);
}
```

```
void ReadCycle()
{
  for (int ReadByte=0; ReadByte<7; ReadByte++){
  //  delay(10);
    for (int bitCount=0;bitCount<10;bitCount++){ //10 bit byte
      int  var = 0;
      while(var <2){ // 2 phase per clock cycle
   currentMicros = micros();
   if (currentMicros - previousClkMicros > ClkTime){ // change state every 550uS
        if (clkState==LOW){
       delayMicroseconds(50); // time for wifi overhead
           clkState = HIGH;
           digitalWrite(TxClock,HIGH);
           previousClkMicros = currentMicros;

   }
else {        //clkState was HIGH

 if (bitCount >0 && bitCount <8){ // strip Start,top 3 bits, Stop, and parity
         delayMicroseconds(50); // wifi overhead
         val = digitalRead(RxData); // Read at end of HIGH
         bitWrite(meterByte[ReadByte], bitCount-1, val);// write bit state
  }
           clkState = LOW;
           digitalWrite(TxClock,LOW);
           previousClkMicros = currentMicros;
      }
       var++;
    }
  }
}
}
 Serial.println();
}
```

```
void PreClock(){  // 50 ascii char pre clock

 for (int y=0; y<30; y++){ //NOT 75, 10, 20

    for (int i=0;i<10;i++){ // ascii char 1S, 7Db, 1P, 1ST = 10 bits
      int  var = 0;
      while(var <2){
   currentMicros = micros();
   if (currentMicros - previousClkMicros > ClkTime){ // change state every 550uS

        if (clkState==LOW){

           clkState = HIGH;
           digitalWrite(TxClock,HIGH);
           previousClkMicros = currentMicros;
   }
else {        //clkState was HIGH

           clkState = LOW;
           digitalWrite(TxClock,LOW);
           previousClkMicros = currentMicros;
      }
       var++;
    }
  }
}
}
}
```

```
void DataPrint(){

  for (int i=1; i<6; i++){

 char ab = meterByte[i];
 Serial.print(ab);
}
Serial.print(".");

 char ab = meterByte[6];
 Serial.print(ab);

 Serial.print(" cycle count is  ");

 cycle++;
 Serial.println(cycle); // counts between resets
}
```

```
void SyncCycle() // potentially loose sync after 25 or so - need to renull if R is not found
{
  for (int ReadByte=0; ReadByte<36; ReadByte++){  // 36 bytes
    for (int bitCount=0;bitCount<10;bitCount++){ // 10 bits
      int  var = 0;
      while(var <2){
   currentMicros = micros();
   if (currentMicros - previousClkMicros > ClkTime){ // change state every 500uS
        if (clkState==LOW){
           clkState = HIGH;
           digitalWrite(TxClock,HIGH);
           previousClkMicros = currentMicros;
   }
else {        //clkState was HIGH
  if (bitCount >0 && bitCount <8){ // strip Start, Stop, and parity
          val = digitalRead(RxData); // Read on HIGH
          bitWrite(meterByte[ReadByte], bitCount-1, val);// write bit state
  }
           clkState = LOW;
           digitalWrite(TxClock,LOW);
           previousClkMicros = currentMicros;
      }
       var++;
    }
  }
}
char ab = meterByte[ReadByte];

if ((meterByte[ReadByte])== 82){  //R=82  This works!!!
  break;
   }
 }
}
```

```
void AlignByte(){//in 360 bits find low

      for (int bitCount=0; bitCount<360; bitCount++){
      int  var = 0;
      while(var <2){
   currentMicros = micros();
   if (currentMicros - previousClkMicros > ClkTime){ // change state every 500uS

        if (clkState==LOW){

           clkState = HIGH;
           digitalWrite(TxClock,HIGH);
           previousClkMicros = currentMicros;
   }
else {        //clkState was HIGH

          val = digitalRead(RxData); // Read on HIGH
         if (val==LOW){
           break;
          }

           clkState = LOW;
           digitalWrite(TxClock,LOW);
           previousClkMicros = currentMicros;

      }
       var++;
    }
  }
 }
}
```

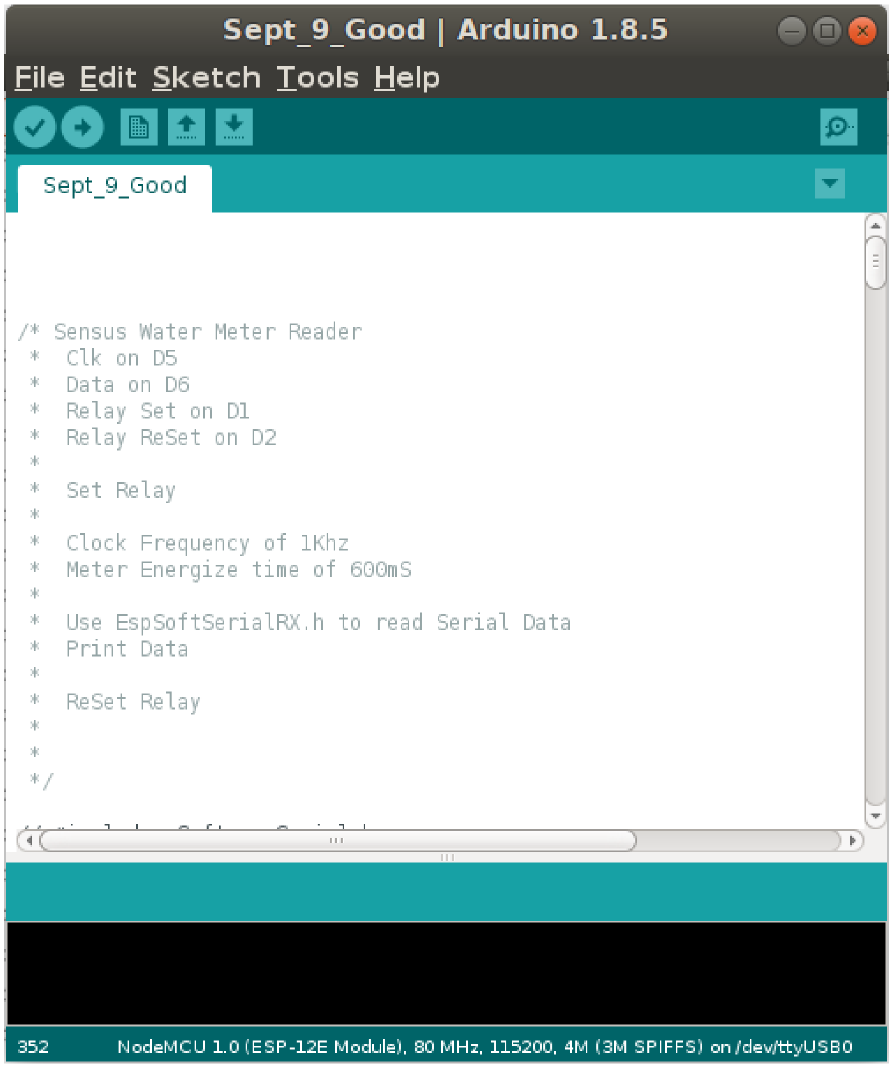
```
void FindNull(){   //look for 11 ones in a row
  int eleven = 0;
      for (int bitCount=0; bitCount<360; bitCount++){
      int  var = 0;
      while(var <2){
   currentMicros = micros();
   if (currentMicros - previousClkMicros > ClkTime){ // change state every 500uS
        if (clkState==LOW){
           clkState = HIGH;
           digitalWrite(TxClock,HIGH);
           previousClkMicros = currentMicros;
   }
else {        //clkState was HIGH
          val = digitalRead(RxData); // Read at end of HIGH
         if (val==HIGH){
                       eleven++;
                       if (eleven ==11){
           break;
               }
           }
  else { // val == LOW

    eleven = 0;
  }
           clkState = LOW;
           digitalWrite(TxClock,LOW);
           previousClkMicros = currentMicros;
      }
       var++;
    }
  }
 }
}
```

```
void GetData(){
    PreClock(); // 0-50 ascii char
    FindNull(); // start in a null patch
    AlignByte(); // look for start bit within 45 bytes
    SyncCycle(); //collect bytes until B
    ReadCycle(); // Get 5 digit reading

}
```

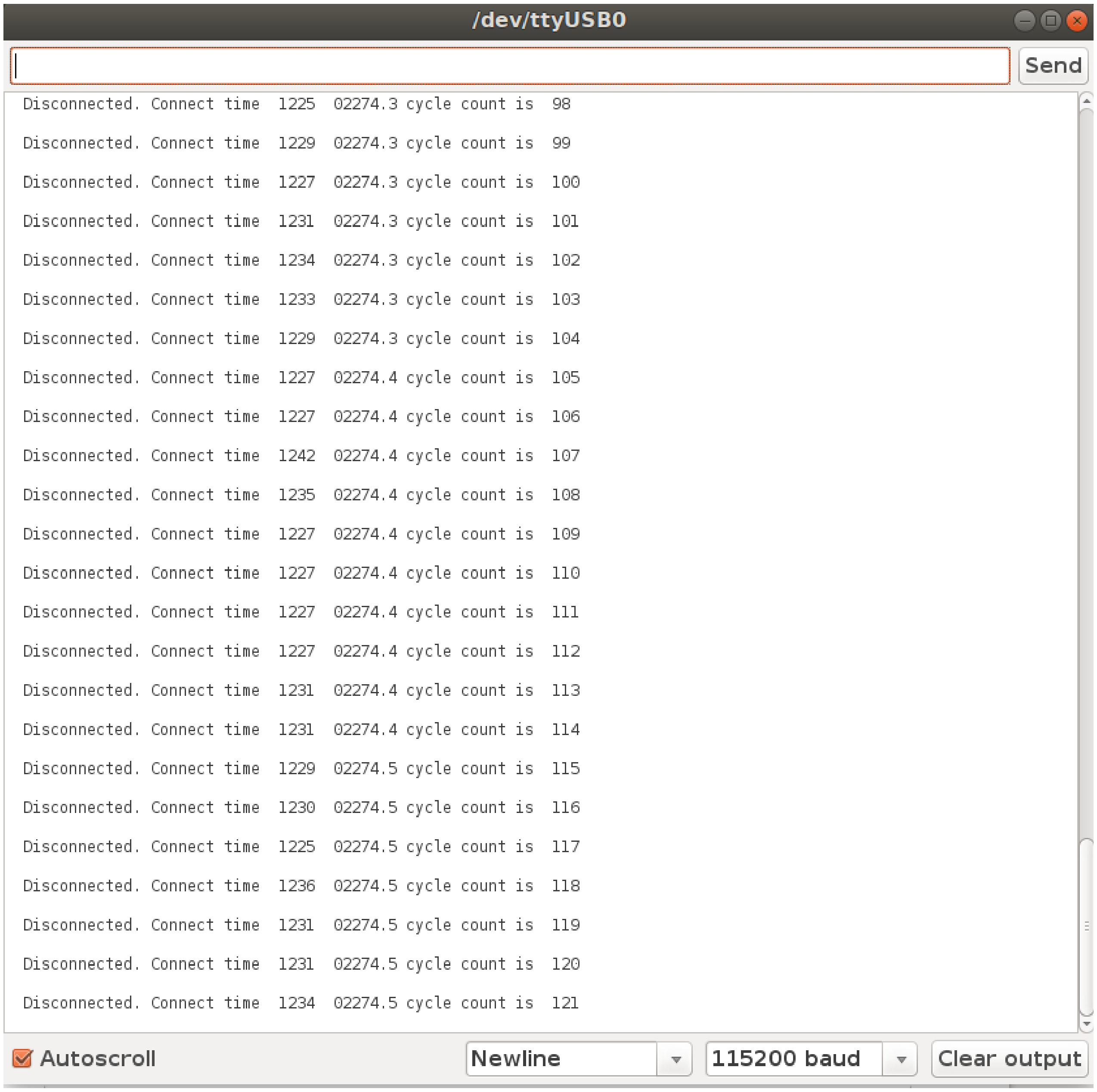
Run Arduino sketch under IDE / Go to Tools / Monitor window



```
/* Sensus Water Meter Reader
 *  Clk on D5
 *  Data on D6
 *  Relay Set on D1
 *  Relay ReSet on D2
 *
 *  Set Relay
 *
 *  Clock Frequency of 1Khz
 *  Meter Energize time of 600mS
 *
 *  Use EspSoftSerialRX.h to read Serial Data
 *  Print Data
 *
 *  ReSet Relay
 *
 *
 */
```

352 NodeMCU 1.0 (ESP-12E Module), 80 MHz, 115200, 4M (3M SPIFFS) on /dev/ttyUSB0

Leave this window open



```
Disconnected. Connect time  1225  02274.3 cycle count is  98
Disconnected. Connect time  1229  02274.3 cycle count is  99
Disconnected. Connect time  1227  02274.3 cycle count is  100
Disconnected. Connect time  1231  02274.3 cycle count is  101
Disconnected. Connect time  1234  02274.3 cycle count is  102
Disconnected. Connect time  1233  02274.3 cycle count is  103
Disconnected. Connect time  1229  02274.3 cycle count is  104
Disconnected. Connect time  1227  02274.4 cycle count is  105
Disconnected. Connect time  1227  02274.4 cycle count is  106
Disconnected. Connect time  1242  02274.4 cycle count is  107
Disconnected. Connect time  1235  02274.4 cycle count is  108
Disconnected. Connect time  1227  02274.4 cycle count is  109
Disconnected. Connect time  1227  02274.4 cycle count is  110
Disconnected. Connect time  1227  02274.4 cycle count is  111
Disconnected. Connect time  1227  02274.4 cycle count is  112
Disconnected. Connect time  1231  02274.4 cycle count is  113
Disconnected. Connect time  1231  02274.4 cycle count is  114
Disconnected. Connect time  1229  02274.5 cycle count is  115
Disconnected. Connect time  1230  02274.5 cycle count is  116
Disconnected. Connect time  1225  02274.5 cycle count is  117
Disconnected. Connect time  1236  02274.5 cycle count is  118
Disconnected. Connect time  1231  02274.5 cycle count is  119
Disconnected. Connect time  1231  02274.5 cycle count is  120
Disconnected. Connect time  1234  02274.5 cycle count is  121
```

Autoscroll [✓] · Newline · 115200 baud · Clear output