```
// 230322_Sounder_Timer0_pulses for Seafarer
// Trigger TR10 with a pulse 2 times a second via pin 5
// Messure time between pulse and echo on pin 2
// Display depth with PWM voltage via pin 3
// 1500 m/s = 1.5 m/millis or 1.500 mm/micro

#include <SoftwareSerial.h>
#include <SD.h>
SoftwareSerial mySerial(8,7);
#define chipSelect 10
File logfile;

#include <Adafruit_GPS.h>
#include <SPI.h>
Adafruit_GPS GPS(&mySerial);
#define LOG_FIXONLY false

const int pulseOutPin = 5;
const int displayPin = 3;
const int pulseInPin = 2; //
const int vSound = 1480; // m/s
const int pulseOutDuration = 20; //microseconds
const int timeBetweenPulsea = 500; // milliseconds
float volt = 0.0;
unsigned long previousMillis;  //violate

void Update(unsigned long currentMillis)
{
 if(currentMillis - previousMillis >= timeBetweenPulsea)
 {
  previousMillis = currentMillis;

  // send a pulse to the sounder
  digitalWrite(pulseOutPin, HIGH);
  delayMicroseconds(pulseOutDuration);
  digitalWrite(pulseOutPin,LOW);

  // messure depth and display
  unsigned long travelTime = pulseIn(pulseInPin, LOW) + 300;
  // 300 is pulselenght at mode 6x
  unsigned long depth = travelTime * vSound / 20000; //cm
  // devide depth in a way thay the volts value equals meters
  volt = depth/ 1.55; // calibrated value at depth 110
  // maximum displayed depth is 5 * 1.93 = 9.65
  if (volt >= 255)
    volt = 255;
  analogWrite(displayPin,volt);

  // get $GPRMC from GPS
  if (GPS.newNMEAreceived())
  {
   char *stringptr = GPS.lastNMEA();
```

```
    if (!GPS.parse(stringptr))
      return;
    if (LOG_FIXONLY && !GPS.fix)
      return;
    //write the string to the SD file
    uint8_t stringsize = strlen(stringptr);
    if (stringsize != logfile.write((uint8_t *)stringptr, stringsize))
      Serial.println("error");
    if (strstr(stringptr, "RMC"))
    {
      unsigned long travelTime = pulseIn(2,LOW) + 300;
      unsigned long depth = travelTime * vSound / 20000;
      logfile.println();
      logfile.print("$SDDBT,");
      logfile.println(depth);
      logfile.println(millis());
      logfile.println();
      logfile.flush();
    }
   }
  }
}

void setup()
{
  pinMode (pulseOutPin, OUTPUT);
  pinMode (pulseInPin, INPUT);
  pinMode (displayPin, OUTPUT);
  pinMode(chipSelect, OUTPUT);
  if (!SD.begin(chipSelect))       // see if the card is present and can be initialized:
    Serial.println("Card init. failed!");
  char filename[15];
  strcpy(filename, "RMCDPT00.TXT");
  for (uint8_t i = 0; i < 100; i++)
  {
    filename[6] = '0' + i/10;
    filename[7] = '0' + i%10;
    // create if does not exist, do not open existing, write, sync after write
    if (! SD.exists(filename))
    {
      break;
    }
  }
  logfile = SD.open(filename, FILE_WRITE);
  GPS.begin(9600);
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_5HZ);  // 5 times per second
  GPS.sendCommand(PMTK_API_SET_FIX_CTL_5HZ);
  GPS.sendCommand(PGCMD_NOANTENNA);
  OCR0A = 0xAF;
  TIMSK0 |= _BV(OCIE0A);
}
```

```
// Interrupt is called once a millisecond
SIGNAL(TIMER0_COMPA_vect)
{
  char c = GPS.read();
  #ifdef UDR0              //USART I/O Data Register
    if (c) UDR0 = c;
  #endif
  unsigned long currentMillis = millis();
  Update(currentMillis);
}

void loop()
{
}
```

```
// 230322_shield_sdlog
// Saving $GPRMC string and depth on SDcard

#include <SPI.h>
#include <Adafruit_GPS.h>
#include <SoftwareSerial.h>
#include <SD.h>
#include <avr/sleep.h>

SoftwareSerial mySerial(8, 7);
Adafruit_GPS GPS(&mySerial);
#define chipSelect 10
File logfile;
const int vSound = 1480; // m/s


void setup()
{
  Serial.begin(115200);
  pinMode(chipSelect, OUTPUT);
  if (!SD.begin(chipSelect))      // see if the card is present and can be initialized:
    Serial.println("Card init. failed!");
  char filename[15];
  strcpy(filename, "RMCDPT00.TXT");
  for (uint8_t i = 0; i < 100; i++)
  {
    filename[6] = '0' + i/10;
    filename[7] = '0' + i%10;
    // create if does not exist, do not open existing, write, sync after write
    if (! SD.exists(filename))
    {
      break;
    }
  }
  logfile = SD.open(filename, FILE_WRITE);
  GPS.begin(9600);
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_5HZ);  // 5 times a second
  GPS.sendCommand(PMTK_API_SET_FIX_CTL_5HZ);
  GPS.sendCommand(PGCMD_NOANTENNA);
  OCR0A = 0xAF;           // Output Compare Register A
  TIMSK0 |= _BV(OCIE0A);  // Timer/Counter Interrupt Mask Register
}

SIGNAL(TIMER0_COMPA_vect)   // Interrupt is called once a millisecond.
{                       // writing direct to UDR0
  char c = GPS.read();
  #ifdef UDR0             //USART I/O Data Register
    if (c) UDR0 = c;
  #endif
}
```

```
void loop()
{
  if (GPS.newNMEAreceived())
  {
    char *stringptr = GPS.lastNMEA();
    if (!GPS.parse(stringptr))
      return;
    uint8_t stringsize = strlen(stringptr);
    if (stringsize != logfile.write((uint8_t *)stringptr, stringsize))
       Serial.println("ërror");
    if (strstr(stringptr, "RMC"))
    {
      unsigned long travelTime = pulseIn(2,LOW) + 300.0;
      unsigned long depth = travelTime * vSound / 20000.0;
      logfile.println();
      logfile.print("$SDDBT,");
      logfile.println(depth);
      logfile.println();
      logfile.flush();
    }
  }
}
```