Compte rendu TP GPS

Cyrile Delattre

Valentin Bouet

Alexandre Malter

SN2 - DOSSIER N°1

Système : SONDEUR MARIN

Fiche résumée

Le système de sondeur marin permet :

- D'obtenir via un GPS la position du bateau (non utilisé ici).
- D'obtenir la T° de l'eau
- D'obtenir la profondeur d'eau sous le bateau
- D'obtenir et de déterminer s'il y a des êtres vivants entre le fond du bateau et le sol marin

Nous ne nous intéressons ici qu'au module de sondeur et à la liaison série de celui-ci. Le but étant de déterminer la partie de la trame NMEA 183 qui répond à nos besoins. En effet, le sondeur des sa mise en route envoi en permanence les données sur le port série. Charge à l'application sur le PC d'extraire les parties importantes.

TP1 :

Acquisitions fondamentales
Etude globale du système
Etude de la liaison série et des trames de la norme NMEA 183

Analyse et validation des savoirs
Lecture des trames du sondeur via RS232

Recherche et synthèse
Recherche d'un mode d'extraction des informations utiles de la trame NMEA 183.
Codage et tests.

Documents fournis :
Documentation technique du sondeur
API win32 (ftp en cours dans la section)

Sondeur Marin

RS232

Acquisition des trames et traitement

GRILLE DE NOTATION – TP SYSTEMES

NOMS DES ETUDIANTS		Application vérifiée par :	Date
TITRE DU TP :			

QUESTIONS PRELIMINAIRES	**NOTATION**	**/ 4**
Précision et pertinence des réponses	**/4**	
Qualité de la rédaction (présentation, orthographe, grammaire,…)	Bonus 1pt	

COMPTE RENDU	**NOTATION**	**/ 6**
Sommaire, but (obligatoire sinon perte de points)	**-1 pt possible**	
Principe	**/3**	
Copies d'écrans avec explications(obligatoire sinon perte de points)	**-1 pt possible**	
Conclusion (problèmes rencontrés, résolution, vécu du TP, proposition d'évolution)	**/2**	
Qualité de la rédaction du compte rendu	**/1**	

APPLICATION	**NOTATION**	**/ 10**
Classe(s), (Réutilisabilité, complétude, Utilisation, syntaxe)	**/3**	
Qualité du code (entête et commentaires, Qualité d'implémentation)	**/3**	
Correspondance avec les objectifs (recettage)	**/3**	
Capacité à réaliser tout leTP	**/1**	
	TOTAL /20	

COMMENTAIRES / REMARQUES / CONSEILS :

Sommaire

-but……..page 4

-principe……page 4

-questions……page 5

-algorithme……page 6

-programme C++……..page 7-11

-conclusion……page 12

But :

Etude globale du système Etude de la liaison série et des trames de la norme NMEA 183

Lecture des trames du sondeur via RS232

Recherche et synthèse Recherche d'un mode d'extraction des informations utiles de la trame NMEA 183.

Codage et tests.

Principe :

Nous avons effectué des recherches sur les trame NMEA 183

Après ça, une application a été développée pour via un le port série du PC afficher la trame reçue dans un mémo, pour une durée d'acquisition de 15s

L'application doit aussi afficher la température de l'eau, et la profondeur en continue et afficher la position courante du sondeur.

LOWRANCE
www.lowrance.com

Pub. 988-0152-021

M56 S/Map

Fish-finding Sonar & Mapping GPS

Installation and Operation Instructions

Questions Préliminaires

1) **Donnez la distance max entre 2 matériels connectés par liaison RS232**

   La distance max entre 2 matériels connectées est de 15 Mètre pour une liaison RS232.

2) **Donnez les caractéristiques de transmission (vitesse…) du sondeur/GPS**

   Le sondeur GPS a pour caractéristiques d'avoir un sonar avec une vitesse de transmission de 1.480 m/s et un GPS de 300 000 Km/s de vitesse de transmission.

3) **Expliquez les portions de la trame NMEA 183 qui permettent d'obtenir la température et la profondeur de l'eau.**

```
$GPGGA       : Type de trame
064036.289   : Trame envoyée à 06 h 40 min 36 s 289 (heure UTC)
4836.5375,N  : Latitude 48,608958° Nord = 48° 36' 32.25" Nord
00740.9373,E : Longitude 7,682288° Est = 7° 40' 56.238" Est
1            : Type de positionnement (le 1 est un positionnement GPS)
04           : Nombre de satellites utilisés pour calculer les coordonnées
3.2          : Précision horizontale ou HDOP (Horizontal dilution of precision)
200.2,M      : Altitude 200,2, en mètres
,,,,,0000    : D'autres informations peuvent être inscrites dans ces champs
*0E          : Somme de contrôle de parité, un simple XOR sur les caractères entre
$ et *[3]
```

```
$GPRMC,053740.000,A,2503.6319,N,12136.0099,E,2.69,79.65,100106,,,A*53

$GPRMC       : type de trame
053740.000   : heure UTC exprimée en hhmmss.sss : 5 h 37 min 40 s
A            : état A=données valides, V=données invalides
2503.6319    : Latitude exprimée en ddmm.mmmm : 25° 03.6319' = 25° 03' 37,914"
N            : indicateur de latitude N=nord, S=sud
12136.0099   : Longitude exprimée en dddmm.mmmm : 121° 36.0099' = 121° 36' 00,594"
E            : indicateur de longitude E=est, W=ouest
2.69         : vitesse sur le fond en nœuds (2,69 nd = 3,10 mph = 4,98 km/h)
79.65        : route sur le fond en degrés
100106       : date exprimée en qqmmaa : 10 janvier 2006
,            : déclinaison magnétique en degrés (souvent vide pour un GPS)
,            : sens de la déclinaison E=est, W=ouest (souvent vide pour un GPS)
A            : mode de positionnement A=autonome, D=DGPS, E=DR
*53          : somme de contrôle de parité au format hexadécimal[3]
```

De base nous ne pouvons pas récupérer la température et la profondeur de l'eau il faut modifier la trame et l'adapter.

4) **Quelle(s) fonction(s) de traitement de chaîne de caractères vous utiliser pour séparer les données dont vous avez besoins et celles inutiles pour vous ?**

La fonction utiliser est « snprintf » une fonction utile pour découper des tableaux.

5) **Expliquez ce que sont la latitude et la longitude ?**

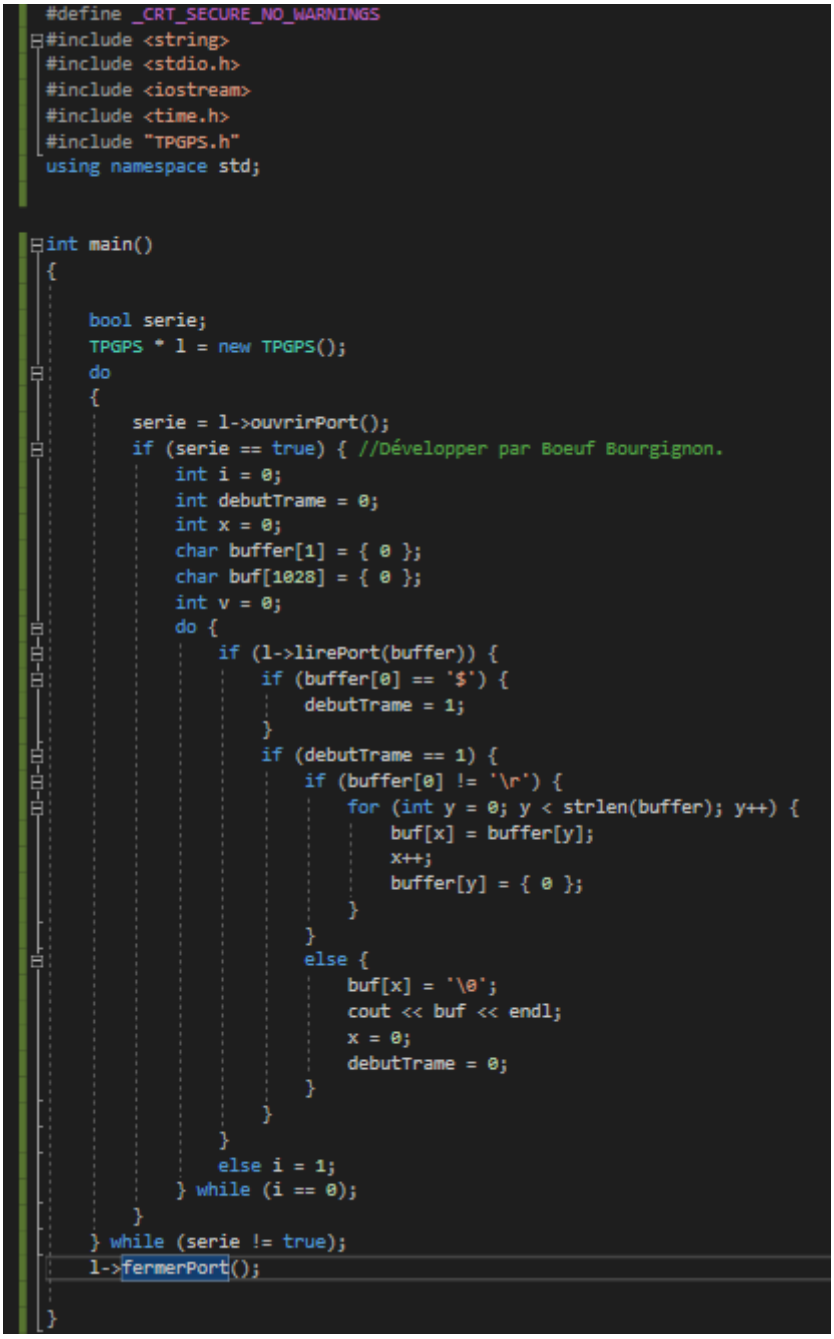
La ***latitude*** est une valeur angulaire, expression du positionnement nord ou sud d'un point sur Terre. D'un point de vue mathématique, la latitude d'un point est l'angle au centre que forme la normale (verticale) en ce point avec le plan équatorial.

La ***longitude*** est une valeur angulaire, expression du positionnement est ou ouest d'un point sur Terre. En géodésie, c'est l'angle au centre que forme le plan passant par ce point et par l'axe de rotation de la terre avec le plan du méridien de Greenwich.

Programme C++

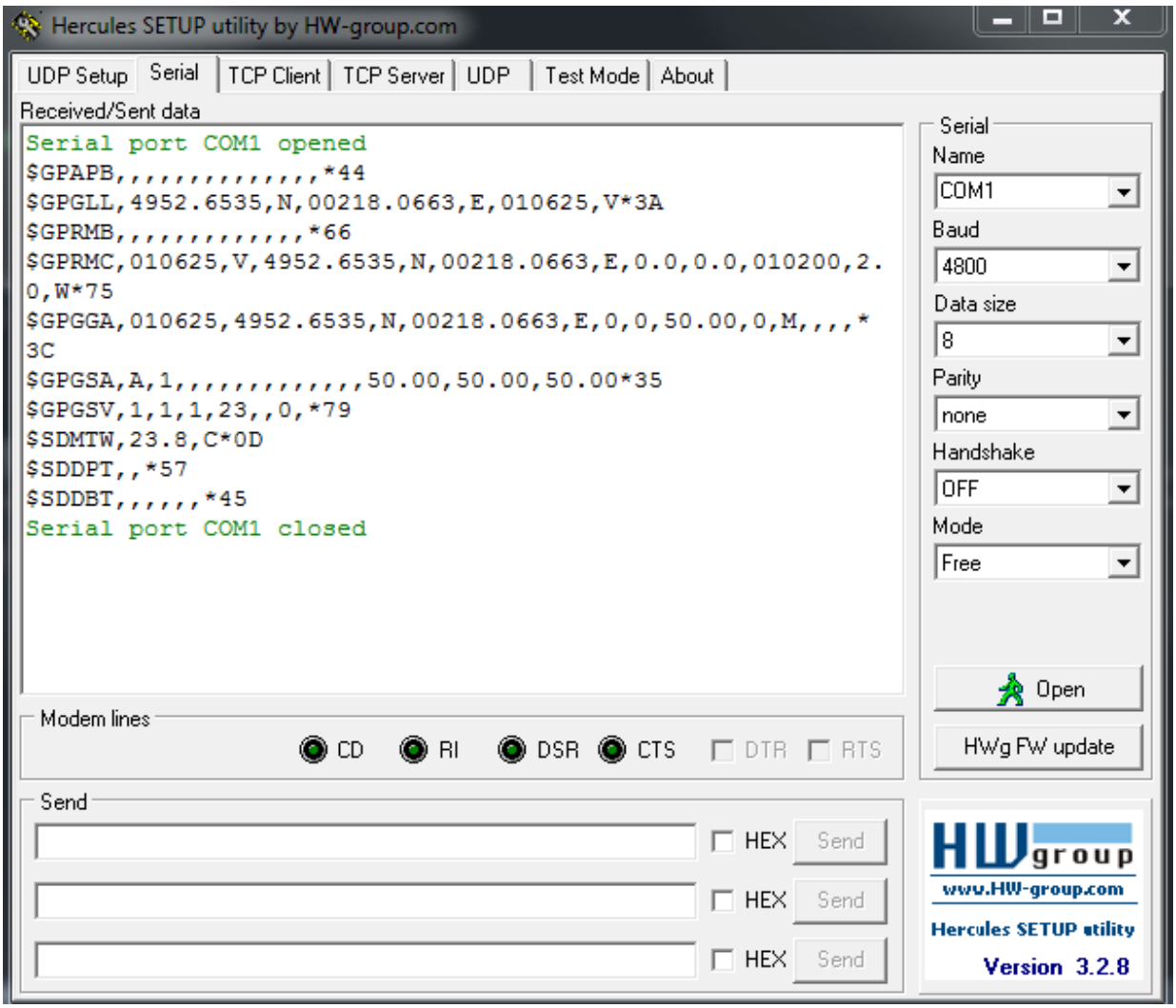
```cpp
#include "TPGPS.h"

TPGPS::TPGPS()
{

}
// fonction qui ouvre et se connecte au port série du GPS.
bool TPGPS::ouvrirPort()
{
    hcom = CreateFileA("COM1", GENERIC_READ | GENERIC_WRITE, 0, NULL, OPEN_EXISTING, FILE_FLAG_NO_BUFFERING, NULL);
    if (hcom == INVALID_HANDLE_VALUE)
    {
        return false;
    }
    else
    {
        GetCommState(hcom, &dcb);
        dcb.BaudRate = CBR_4800;
        dcb.ByteSize = 8;
        dcb.Parity = NOPARITY;
        dcb.StopBits = ONESTOPBIT;

        SetCommState(hcom, &dcb);
        com.ReadIntervalTimeout = MAXDWORD;
        com.ReadTotalTimeoutMultiplier = 0;
        com.ReadTotalTimeoutConstant = 0;
        com.WriteTotalTimeoutMultiplier = 0;
        com.WriteTotalTimeoutConstant = 0;

        SetCommTimeouts(hcom, &com);

        return true;
    }

}
// fonction pour récupérer les trames GPS.
bool TPGPS::lirePort(char * buffer)
{
    int n = 0;
    unsigned long d = 0;
    if ((n = ReadFile(hcom, buffer, 1, &d, NULL)) < 0)
    {
        return false;
    }
    buffer[n] = '\0';
    return true;
}
// fonction pour écrire sur le port série (pas utilse pour ce projet).
bool TPGPS::ecrirePort()
{
    return true;
}
// fonction pour fermer le port série.
void TPGPS::fermerPort()
{
    CloseHandle(hcom);
}
```

Ci-dessus, la configuration du port série pour permettre une communication entre le Form et le GPS. On retrouve le port dans CreateFile (dans notre cas le COM6), et la suite le nombre de baud, la nombre de bit, la parité et le bit de stop.

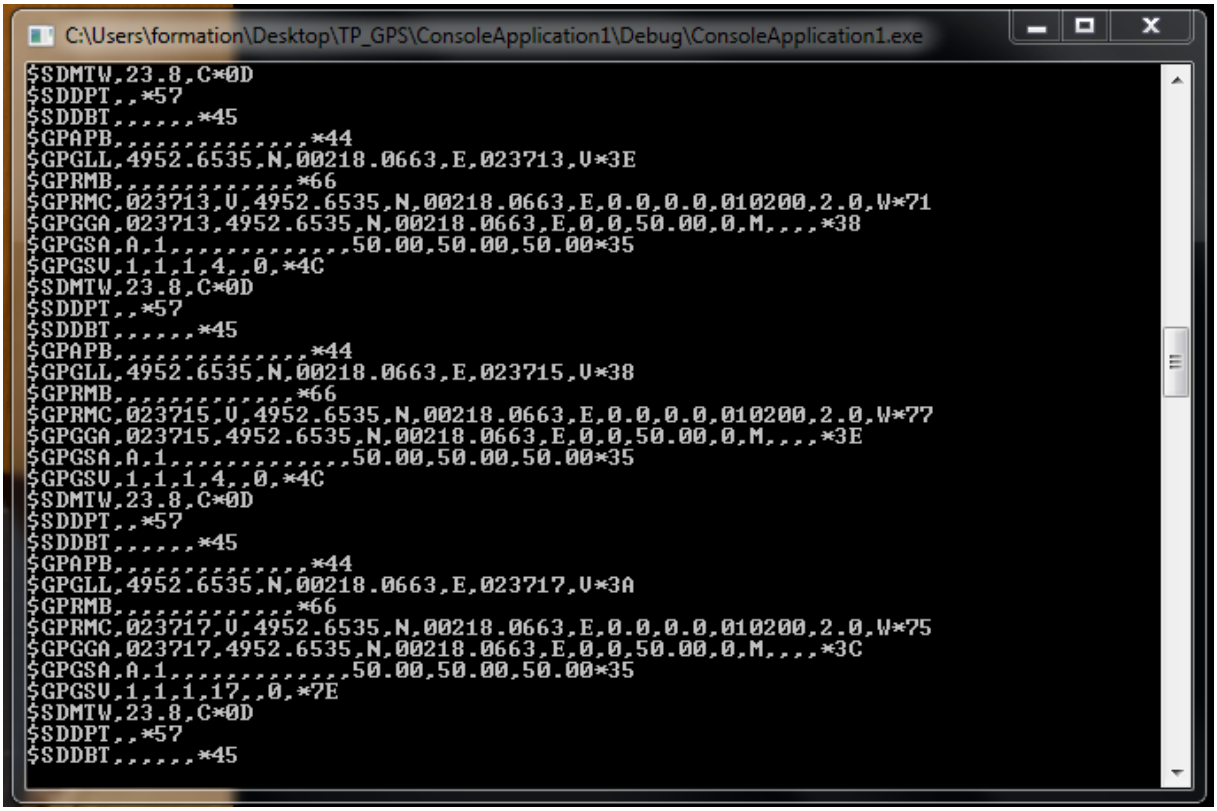


```
#define _CRT_SECURE_NO_WARNINGS
#include <string>
#include <stdio.h>
#include <iostream>
#include <time.h>
#include "TPGPS.h"
using namespace std;


int main()
{

    bool serie;
    TPGPS * l = new TPGPS();
    do
    {
        serie = l->ouvrirPort();
        if (serie == true) { //Développer par Boeuf Bourgignon.
            int i = 0;
            int debutTrame = 0;
            int x = 0;
            char buffer[1] = { 0 };
            char buf[1028] = { 0 };
            int v = 0;
            do {
                if (l->lirePort(buffer)) {
                    if (buffer[0] == '$') {
                        debutTrame = 1;
                    }
                    if (debutTrame == 1) {
                        if (buffer[0] != '\r') {
                            for (int y = 0; y < strlen(buffer); y++) {
                                buf[x] = buffer[y];
                                x++;
                                buffer[y] = { 0 };
                            }
                        }
                        else {
                            buf[x] = '\0';
                            cout << buf << endl;
                            x = 0;
                            debutTrame = 0;
                        }
                    }
                }
                else i = 1;
            } while (i == 0);
        }
    } while (serie != true);
    l->fermerPort();

}
```

Ci-dessus, le code permettant e trouver la trame pour récupérer la température (on a supposé que c'était celle de la température) + affichage avec snprintf.



Les trames NMEA entières envoyé par le GPS qui ont été récupérés grâce à Hercules pour vérifier les différents paramètres à rentrer et les différentes trames émises.



Ci-dessus, affichage en console après compilation du projet, on voit bien qu'il récupère les trames voulu (longitude, latitude et température) à intervalle de temps régulier.