Wrocław, 25 listopada 2019r.

Mikołaj Baran, 241128
Jakub Aniszewski, 241133

prowadzący: Dominik Żelazny

Laboratorium Urządzeń Peryferyjnych

Ćwiczenie 7 - GPS

1 Cel ćwiczenia

1. Zapoznać się z zestawem GPS oraz podłączyć via Bluetooth
2. W ramach testu podłączyć GPS na ustawieniach testowych oraz utworzyć połączenie przy użyciu HyperTerminala.
3. Odczytać uzyskane komendy oraz podzielić je według typów wiadomości.
4. Sprawdzić ważność uzyskanych danych i przedyskutować wynik.
5. Napisać program w dowolnym środowisku obiektowym, który będzie obsługiwał transmisję szeregową oraz pozwoli na czytelne przedstawienie uzyskanych danych.
6. Napisać program, który na podstawie samodzielnie uzyskanych danych lub od prowadzącego (plik tekstowy, format NMEA) zlokalizuje na mapie świata (np. z Google Map) punkty, w których znajdowało się urządzenie.

2 Wstęp

GPS (Global Positioning System) -jest systemem nawigacji satelitarnej, który został stworzony przez Departament Obrony Stanów Zjednoczonych. System ten obejmuje swoim zasięgiem całą kulę ziemską. Działanie GPS polega na mierzeniu czasu dotarcia sygnału radiowego z satelitów do odbiornika. Znając prędkość fali elektromagnetycznej oraz dokładny czas wysłania danego sygnału, można obliczyć odległość odbiornika od satelitów. Sygnał dociera do użytkownika na dwóch możliwych częstotliwościach nośnych:

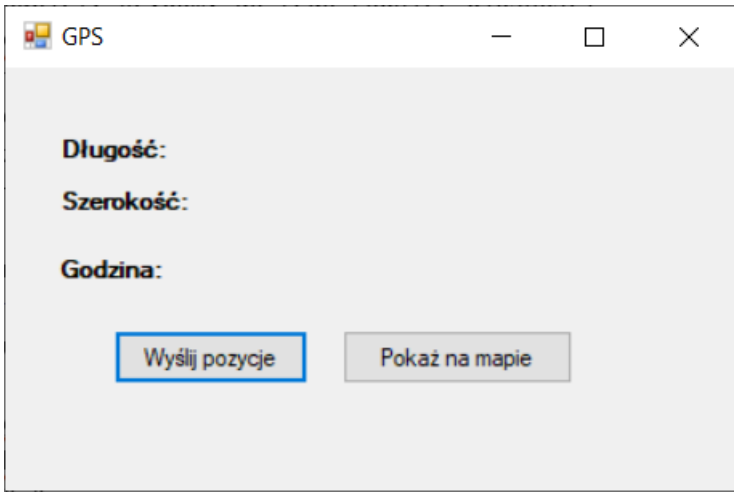
- L1 = 1575,42 MHz (długość fali 19,029 cm)
- L2 = 1227,6 MHz (długość fali 24,421 cm)

Identyfikacja satelitów oparta jest na metodzie podziału kodu **CDMA** (Code Division Multiple Access). Oznacza to, że wszystkie satelity emitują na tych samych częstotliwościach, jednak nadawane sygnały są modulowane innymi kodami. Aby określić pozycję w przestrzeni i czasie, konieczny jest jednoczesny odbiór z co najmniej czterech satelitów. Odbiornik oblicza trzy pseudo-odległości do satelitów oraz odchyłki czasu (różnicy między mało dokładnym dokładnym wzorcem kwarcowym zainstalowanym na odbiorniku oraz bardzo precyzyjnym zegarem atomowym na satelicie). Satelita transmituje w depeszy nawigacyjnej m.in. czas, almanach (stan konstelacji satelitów) oraz efemerydy (parametry lotu satelity). Dzięki tym danym odbiornik GPS jest w stanie określić dokładne współrzędne satelity, w momencie nadania sygnału, co przekłada się następnie na ,przy pomocy pseudo-odległości, na obliczenie własnej pozycji.

3 Opis programu

Program był pisany w języku C# w środowisku Visual Studio. W celu weryfikacji wyłania danych użyty był program PuTTy oraz w celu wyświetlenia naszej pozycji na mapie Google Maps.

Interfejs programu nie jest skomplikowany. Przycisk *Wyślij pozycję* odczytuje aktualną pozycję GPS i wyświetla długość i szerokość geograficzną oraz godzinę pobrania danych. *Pokaż na mapie* pozwala na otwarcie domyślnej przeglądarki i wyświetlenie aktualnej pozycji na mapach Google.



4 Najważniejsze funkcje w programie

4.1 Funkcja otwierająca port z którego odczytywane są dane.

```
private void button1_Click(object sender, EventArgs e)
{

    Thread readThread = new Thread(Read);

    serialPort = new SerialPort();
    serialPort.PortName = "COM3";
    serialPort.BaudRate = 9600;
    serialPort.Parity = Parity.None;
    serialPort.DataBits = 8;
    serialPort.StopBits = StopBits.One;
    serialPort.Handshake = Handshake.None;

    serialPort.ReadTimeout = 500;
    serialPort.WriteTimeout = 500;

    serialPort.Open();
    next = true;
    readThread.Start();
}
```

Na początku tworzymy nowy wątek dzięki któremu będziemy mogli niezależnie od działania programu odczytywać dane. Następnie towrzymy instancję obiektu **SerialPort** z bibliioteki System.IO.Ports oraz ustawiamy wszystkie najważniejsze parametry. Pola **ReadTimeout** oraz **WriteTimeout** pobierają lub ustawiają liczbę milisekund przed upływem limitu czasu, gdy operacja odczytu nie zostanie zakończona. Na koniec otwieramy port i rozpoczynamy wątek.

4.2 Funkcja która czyta dane z otwartego portu.

```
public void Read()
{
    while (next)
    {
        try
        {
            string message = serialPort.ReadLine();
            if (message.Contains("GPGGA"))
            {
                Position position_N = ConvertToCoords(message.Split(',')[2], message.Split(',')
2);
                Position position_E = ConvertToCoords(message.Split(',')[4], message.Split(',')
3);
                GPS.latitudeLabelValue = position_N;
                GPS.longitudeLabelValue = position_E;
                time.Invoke((Action)delegate
                {
                    string timeLocal = message.Split(',')[1];
                    time.Text = timeLocal.Insert(2, ":").Insert(5, ":").Substring(0,timeLocal.I
                    Refresh();
                });
            }
        }
        catch (TimeoutException) { }
    }
}
```

Na początku sprawdzamy czy nasz wątek ma dalej chodzić. Następnie czytamy linię z wiadomością z naszego otwartego portu. Sprawdzamy czy w odczytanej linij znajduje się fraza **GPGGA** czyli Global Positioning System Fix Data ponieważ to te informacje nas interesują oraz to je będziemy przetwarzać. Dalej tworzymy instancje obiektów własnej klasy **Position** przy pomocy funkcji **ConvertToCoords**. Klasa ta zawiera informację o stopniach, minutach, sekundach i typie danych oraz została stworzona w celu ułatwienia manipulacji tymi danymi. Teraz ustawiane są etykiety wyświetlane w GUI oraz przy pomocy delegaty odczytujemy i wyświetlamy czas. Cała manipulacja odczytaną wiadomością odbywa się dzięki metodzie **Split** ponieważ wszystkie dane oddzielone są w wiadomości przecinkami. Dzięki temu wystarczy wiedzieć na której pozycji znajduje się dana informacja.

4.3 Funkcja która zmienia odczytane dane na współrzędne.

```
public static Position ConvertToCoords(string input, string type, int degreeSize = 2)
{
    Position result = new Position();

    result.Degree = int.Parse(input.Substring(0, degreeSize));
    result.Minutes = int.Parse(input.Substring(degreeSize, 2));
    var secs_string = input.Substring(degreeSize + 2, 4);
    double secs_fraction = double.Parse(secs_string,
                        CultureInfo.InvariantCulture.NumberFormat);
    result.Seconds = (int)Math.Truncate(60 * secs_fraction);
    result.Type = type;
    return result;
}
```

Tworzymy instancję klasy **Position** a następnie dzięki temu że dane wysłane są w tej samej formie możemy wpisać je do odpowiednich pól. Na początku z danych bierzemy dwa pierwsze znaki które reprezentują stopnie, parsujemy je na tym **integer** i wpisujemy do pola. To samo robimy

z minutami jednak bierzemy następne dwa znaki. Sekundy na początku parsujemy do typu **double**. Używamy do tego również własności **CultureInfo.InvariantCulture.NumberFormat** ponieważ zawsze chcemy parsować zgodnie z obowiązującym formatem w miejscu w którym jesteśmy a w przypadku urządzenia GPS nasza pozycja może się czesto zmieniać. Na koniec standardowo już parsujemy wartość do typu integer zaokrąglając do najbliższej liczby całkowitej w kierunku zera i zwracamy naszą pozycję.

4.4 Funkcja która pokazuje naszą pozycję na mapie.

```
private void button3_Click(object sender, EventArgs e)
{
    System.Diagnostics.Process.Start("https://maps.google.com/?q=" +
    latitudeLabelValue.CoordsToOneValue.ToString(CultureInfo.GetCultureInfo("en-US")) + "," +
    longitudeLabelValue.CoordsToOneValue.ToString(CultureInfo.GetCultureInfo("en-US")));
}
```

Metoda **Process.Start** uruchamia zasób procesu. jako argument przyjmuje **String** i następnie kojarzy go ze składnikiem. Jako string przesyłamy adres Map Googla poszerzony o odpowiednio sprasowane szerokość i wysokość geograficzną. **CoordsToOneValue** zwraca współrzędne w następującym formacie "Degree + (Minutes / (double)60) + (Seconds / (double)3600)"

5 Wnioski

Napisany przez nas na zajęciach program pozwolił nam na zapoznanie się z możliwościami i sposobami obsługi urządzeń GPS na komputerze. Poznaliśmy również działanie protokołu **NMEA**, powszechnie stosowanym w komunikacji urządzeń elektronicznych. W dalszych krokach ćwiczenia odkodowaliśmy dane wysyłane przez urządzenie GPS na długość i szerokość geograficzną, a także aktualny czas. Udało się również poprawnie sformatować dane, aby wyświetlić aktualną pozycję GPS na Mapach Google.