

project: SIMHUB

Een mini racing simulator



Gemaakt door : Max vd Boom, Ryan vd Broek & Maarten Jakobs
Datum : 22-6-2017
Opleiding : Applicatie- en mediaontwikkelaar
Dank aan : C.Loomans, J.Brandwijk, D.Kalsbeek & J.Dries

Meer informatie:
Wij hebben een Do It Yourself Racing simulator voor Dirt 3 gemaakt. Dit werkt allemaal via De Arduino en een C# applicatie

- Doelen
- Meer kennis over Arduino
 - Meer kennis met C#
 - Project uitwerken
 - Plannen
 - Documenteren
 - Andere indruk geven over ICT

- Gebruikte materialen en gemaakte kosten:
- Arduino
 - 2x TM1638 €10,-
 - Handrem
 - Schakelpook €13,-

```
#region receive
IPEndPoint remoteEndPoint = new IPEndPoint(IPAddress.Parse(I

Thread receiveThread = new Thread(ReceiveData);
receiveThread.IsBackground = true;
receiveThread.Start();

UdpClient client = new UdpClient();

try
{
    string text;
    do
    {
        text = Console.ReadLine();

        if (text.Length != 0)
        {
            //gets the bytes from the game and puts them in
            byte[] data = Encoding.UTF8.GetBytes(text);
            client.Send(data, data.Length, remoteEndPoint);
        }
        while (text.Length != 0);
    }
}

(comPort2Open == true)

#region data to send to second arduino
//this will make sure it starts reading from this line of code
#region garbage filter
dataToSend2[0] = Convert.ToChar("a");
dataToSend2[1] = Convert.ToChar("b");
#endregion

////this will send the data from the totaltime to arduino in
#region Total Time
if (timeArray.Count() > 1)
{
    dataToSend2[2] = timeArray[0];
    dataToSend2[3] = timeArray[1];
    dataToSend2[4] = timeArray[3];
    dataToSend2[5] = timeArray[4];
}
#endregion

//this will send the data from the Speed to arduino in asq
#region Speed
dataToSend2[6] = speedArray[0];
dataToSend2[7] = speedArray[1];
dataToSend2[8] = speedArray[2];
#endregion

//makes a connection with a local server to recive data from
IPEndPoint anyIP = new IPEndPoint(IPAddress.Any, 20777);
byte[] dataGame = client.Receive(ref anyIP);

#region get data out of game
//this gets the round data out of the game and puts it in a
#region Round
float round = BitConverter.ToSingle(dataGame, 144) + 1;
string roundString = Convert.ToString(round);
#endregion

//this gets the gear data out of the game and puts it in a v
Gear

//this gets the speed data out of the game and puts it in a
#region Speed
float speed = BitConverter.ToSingle(dataGame, 28);
string speedString = ((int)Math.Round(speed * 3.6, 0)).ToString("0");
#endregion

//this gets the RPM data out of the game and puts it in a v
#region RPM
float RPM = BitConverter.ToSingle(dataGame, 148);
string RPMString = ((int)(Math.Round(RPM * 10))).ToString("0");
#endregion

if (Serial.available() > 0)
{
    Serial.ReadBytes(m_data, 12);
    if (m_data[0] == 97 && m_data[1] == 98)
    {
        //reset count so that the program will
        count = 0;

        //methods that get information from
        cTime();
        RPM();
        Gears();

        //writes information to the tm1638 display
        module.setDisplayToString(Gear + " "
    }
}
```

Ons Plan

Wij zijn deze proftaak bezig met het maken van een DIY (do it yourself) simulation cockpit. Wij gaan meerder dingen uitwerken zoals displays, handrem, versnellingen, matrix display. Wij gaan de displays gebruiken voor het laten zien van de totale race tijd en daarnaast gaan we de current rondetijd aangeven die wordt gedisplayed op een tm1638. Wij gaan hiernaast nog een tm1638 gebruiken om aan te geven in welke ronde je nu zit en op welke plaats je op het moment zit in de game. Wij gaan zelf een handrem ontwerpen en fabriceren. Daarnaast gaan wij ook zelf een versnellingspook ontwerpen en fabriceren. Wij willen ook proberen om een autodashboard te laten werken om daar de snelheid en toeren te laten zien. Dit gaan wij doen via cambus aansluiting. Deze onderdelen gaan allemaal in sync werken op de game Dirt 3 van codemasters.

FOTO 1: Connectie maken met game

We maken connectie met Dirt 3 met behulp van je locale IP van je pc die bij iedereen hetzelfde is. Dirt 3 gebruik de

port 20777 daar komen bytes doorheen.

FOTO 2: Data vertalen

We krijgen de data in een array met bites die we allemaal omzetten naar leesbare data. In onze applicatie zie je dan deze data continu als er iets gebeurt in het spel.

FOTO 3: Doorsturen naar de Arduino

Alle verzamelde data wordt door gestuurd naar de arduino door middel van een usb connenctie tussen de pc en de arduino. In het begin van het programma moet je zeggen op welke COM de arduino zit. Hier kom je achter door bij: Instellingen -> apparaten. Eerst gaat alles door een "Garbage Filter" die alle overbodige informatie dat perongelijk meekomt verwijderd

FOTO 4: Data sturen naar de TM1638

In het arduino programma gaan we alles weer vertalen en dan gaat alles naar de TM1638. Nu kunnen we alles erop zien.

Voor dit project hebben wij meerdere arduinos en arduino boards gebruikt. Wij hebben hierbij 3 arduinos unos gebruikt voor het aansturen van de displays. Deze displays zijn tm1638, wij hebben deze displays gebruikt omdat ze relatief makkelijk in gebruik zijn omdat er veel libraries aanwezig zijn op het internet. Het programma dat op de arduinos staat hebben wij volledig zelf gecodeerd. Hierbij hebben wij meerdere technieken moeten gebruiken. Wij hebben bijvoorbeeld een connectie moeten maken van arduino naar c#. Deze connectie was relatief simpel te maken met de informatie van projecten die wij eerder het jaar gemaakt hebben. Deze connectie tussen c# was nodig omdat wij de telementry van de game ophalen in een c# console applicatie. Omdat je geen hele arrays kan doorsturen naar arduino in een keer hebben wij elk character moeten opsplitsen naar individuele characters die wij doorsturen via een ASCII methode. Deze methode houdt in dat wij in plaats

van cijfers characters doorsturen die dan makkelijk te zijn converteren naar cijfers. Custom handbrake en Shifter. Voor deze proftaak hebben we ook een handrem en schakelpook gemaakt.

