Redactie Johan Smet ON5EX on5ex@uba.be
Rik Strobbe ON7YD on7yd@uba.be
Jos Warnier ON6WJ on6wj@skynet.be

Technical Info

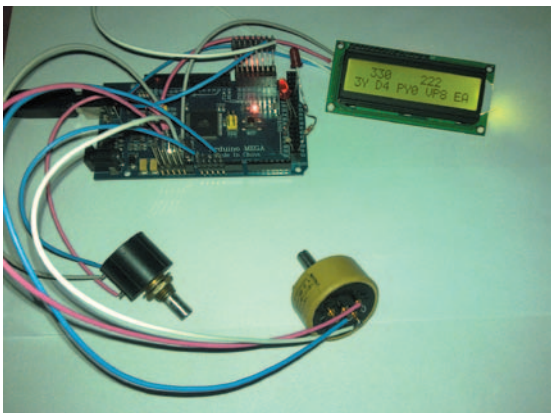
Commande digitale universelle pour moteur d'antenne

Par/door ON4ADN – Traduit par ON5WF

Universele digitale rotorsturing

Avant-propos

J'utilisais une commande analogique pour mon rotor d'antenne home made, mais depuis un certain temps, j'étais à la recherche d'une commande digitale. Lorsque le potentiomètre était hors d'usage, je devais toujours m'arranger pour retrouver le même, pour pouvoir continuer à utiliser l'indication analogique, de 180° à 180°. La commande de rotation de l'antenne avec des transistors à hystérésis ne me plaisait pas, mais à cause de la conception ancienne, il aurait fallu alors renouveler et remonter tout le circuit. Lorsque je lus dans le CQ-QSO que la section WLD avait réalisé un projet de construction d'un afficheur digital pour rotor HAM-4, j'étais à nouveau motivé. Je ne voulais certainement pas me lier à un IC en particulier et qui ne serait plus disponible par la suite (sauf pour un prix élevé). Lorsqu' après quelques jours de recherche sur internet, je tombai sur le nom 'Arduino', j'avais la solution de mon problème. C'était cela: tout 'open source' et en plus:



- Compilateur gratuit (un compilateur est un programme qui traduit en langage machine le programme écrit par le programmeur). Pas besoin de faire appel à un compilateur piraté difficile à obtenir et qui ne marche pas.
- Bibliothèque de programmes gratuite pour le matériel correspondant.
- Explications complètes sur les commandes, beaucoup d'exemples pour expérimenter.
- Beaucoup d'explications sur tout, LCD, entrées, sorties, commande de robots, etc.
- Schémas gratuits etc, pour commander des écrans LCD (bon marché ou chers) avec le software approprié.
- Convient parfaitement pour tout rotor dont la commande est passée en QRT ou devenue introuvable.
- Fichiers gratuits pour réaliser des circuits imprimés pour différents microcontrôleurs ATmega.
- Et encore beaucoup plus…

Tapez "Arduino" dans votre moteur de recherche favori et vous trouverez certainement un exemple pour un futur projet. Même les écrans tactiles sont prévus, de quoi alimenter un projet suivant.

Microcontrôleur utilisé

On utilise ici le microcontrôleur très bon marché ATmega1280. Le circuit que j'ai acheté (un clone) coûte moins de 20 $, livraison à domicile par recommandé.
Les avantages sont nombreux: tout est inclus, y compris le compilateur, bootloader, câble USB/connectique, polyfuse, etc.

Voorwoord

Ik had een analoge sturing in gebruik voor mijn zelfgebouwde antennerotor, maar was al een tijdje op zoek naar een digitale sturing. Als de potmeter buiten defect geraakte, moest ik steeds dezelfde zien te vinden om de analoge uitlezing verder te kunnen gebruiken, 180° naar 180°. De sturing via transistoren met hysteresis om de antenne te draaien beviel mij niet, maar door de ouderwetse opzet zou ik de ganse schakeling moeten vernieuwen en herbouwen. Toen ik in CQ-QSO las dat de sectie WLD een bouwproject had gerealiseerd voor de digitale uitlezing van een HAM-4 rotor, was ik opnieuw gelanceerd. Ik wou me zeker niet vastkluisteren aan een bepaald IC dat later niet meer te krijgen zou zijn (tenzij voor veel centen). Toen ik na enkele dagen zoeken op het internet op een forum op de naam 'Arduino' stuitte, was ik meteen verkocht. Dit was het: alles 'open source' en bovendien

- gratis compiler (een 'compiler' is een programma dat een geschreven programma – de broncode - omzet naar de processoropdrachten). Geen nood aan een moeilijk te verkrijgen of gehackte compiler die niet werkt.
- gratis programmabibliotheek voor de aangesloten hardware
- volledige uitleg over de commando's, veel voorbeelden om uit te proberen
- volledige uitleg over van alles, LCD, ingangen, uitgangen, robotbesturing, enz.
- gratis schema's enz. om (goedkope of dure) LCD's aan te sluiten met bijhorende software
- kan perfect gebruikt worden voor iedere rotor waarvan de sturing QRT is of niet meer te vinden
- gratis files om printen te maken voor verschillende ATmega microcontrollers
- en nog veel meer…

Tik "Arduino" op uw favoriete zoekmachine en je vindt zeker een voorbeeld om verder mee aan de slag te gaan. Zelfs 'on touch displays' zijn geïntegreerd, iets wat ik wel zie zitten voor een volgend project.

Gebruikte micocontroller

Hier wordt gebruik gemaakt van de ATmega1280 microcontroller, die zeer goedkoop te verkrijgen is. Het printje dat ik heb gekocht (een kloon) kost minder dan 20 $, aangetekend thuisgeleverd.
De voordelen zijn legio: alles erop en eraan, inclusief compiler, bootloader, USB kabel/connectie, polyfuse, enz.

Le chip ATmega1280 est du type SMD. Les raisons du choix de ce chip sont les suivantes:

- Je me doutais qu'écrire un programme pour ON (carte DXCC de 180° à 360°/0° et retour à 180°) allait consommer assez bien de mémoire. Le chip 1280 peut stocker environ 128 KB dans la mémoire flash. Le programme utilise environ 80 kB, en ce compris les 4 KB de bootloader. Les chips ATmega de plus bas niveau ne peuvent pas faire cela. Si l'on recherche un affichage de 0° à 360°, le plus petit chip Atmega (Uno) suffit, car le programme occupe particulièrement peu d'espace mémoire. Il suffit alors d'une petite ligne de programme.
- Un chip vierge peut être programmé via le compilateur gratuit (open source).
- Les bootloaders pour les chips Atmega peuvent être programmés sur le chip via le compilateur, ce qui signifie une économie si vous voulez éventuellement passer à un projet avec plusieurs utilisateurs et un print homemade.
- La facture: pour le prix, il n'est pas possible de fabriquer le print complet, sans parler de l'envoi des différentes pièces par la poste.
- Arduino fonctionne avec tous les systèmes d'exploitation: Windows, Linux, Mac,... Juste quelques recherches sur le site et c'est parti.
- Beaucoup d'entrées et de sorties (analogues, digitales, PWM) pour expérimenter.
- Détection automatique de la fourniture de courant, via USB ou alimentation externe.
- Lecture série via le compilateur: on peut suivre sur le PC ce qui se produit après une modification de programme, sans que l'affichage LCD soit raccordé.
- Le chip 1280 (et plus) possède deux sorties spécialement prévues pour piloter des LCD de type I2C.
- Polyfuse intégré (fusible auto réparant) pour protéger le port USB (ou même la carte mère) du PC en cas de “catastrophe”.

Entrées et sorties de la carte Arduino

Le but est de pouvoir employer la commande avec toutes sortes de rotors, en contrôlant 2 sorties digitales qui enclenchent un transistor ou un optocoupleur, lequel à son tour, enclenche un relais ou tout autre élément. Les rotors avec un frein, tels que le HAM-4, sont aussi concernés. Pour activer ou désactiver le frein au moyen d'un relais séparé, avec un délai d'une demi seconde par exemple, il suffit d'ajouter une petite ligne de programme. Il faut alors programmer une sortie supplémentaire, mais il y en a assez.

Je vous conseille, pour toute expérimentation, de placer sur la sortie vers une LED ou tout autre composant, une résistance afin de protéger cette sortie ou même le chip tout entier! Les sorties sont limitées à 40 mA, mais un court circuit est vite produit. Il est aussi conseillé de limiter la puissance fournie par le CHIP (moins il en fournit et mieux c'est pour lui).

Pour mon rotor d'antenne (moteur d'essuie-glace de voiture avec réducteur), j'emploie l'alimentation d'un ancien PC. Assurez-vous que toutes les masses (circuit imprimé et alimentation externe) soient bien raccordées ensembles. Donc, les bornes de masse du circuit imprimé et de l'alimentation externe raccordées au châssis. Il y a plusieurs bornes de masse sur la carte. Jouez la sécurité en en raccordant quelques-unes au châssis.

Deux entrées analogiques sont utilisées pour la lecture de l'état de deux potentiomètres: 1 à l'extérieur, fixé au rotor et 1 à l'intérieur, à la commande, pour choisir la direction. Je voulais un fonctionnement complètement automatique: si le bouton de commande est réglé sur un continent donné, la sortie doit être réglée de telle sorte que l'antenne pointe dans cette direction. Nous employons donc deux entrées analogiques et deux sorties digitales qui enclenchent un relais via un transistor, à gauche ou à droite. Avec un frein de rotor, on utilise trois sorties digitales. Sur la carte, il y a en montage standard sur chaque sortie digitale, une LED SMD qui indique si le circuit est en service (rotor en train de tourner). Il est ainsi possible de vérifier que tout fonctionne, même sans rotor ou relais et voir éventuellement ce qui ne marche pas. Ces LED peuvent aussi servir d'indicateurs.

De ATmega1280 chip is van het SMD-type. De redenen voor de keuze van deze chip zijn:

- ik had het vermoeden dat een programma schrijven voor ON (DXCC-kaart van 180° over 360°/0° en dan terug naar 180°) wel eens behoorlijk veel geheugenruimte zou innemen. De 1280-chip kan ongeveer 128 KB in het flashgeheugen opslaan. Het programma beslaat ongeveer een 80 kB, inclusief de 4 KB bootloader. ‘Lagere’ ATmega chips kunnen dit niet aan. Als je een uitlezing zoekt van 0° naar 360°, voldoet de kleinste ATmega-chip (Uno), want het programma neemt bijzonder weinig ruimte in beslag. Je hoeft dan maar 1 lijntje te programmeren.
- een blanco chip kan via de gratis (open source) compiler geprogrammeerd worden
- bootloaders voor ATmega-chips kunnen via de compiler op de chip geprogrammeerd worden, wat een kostenbesparing betekent als je eventueel overstapt naar een project met meerdere gebruikers en een zelfgemaakte print.
- het kostenplaatje: voor de prijs kan je de volledige print niet zelf maken, laat staan de verschillende stukken via de post laten overkomen.
- Arduino werkt onder alle besturingssystemen: Windows, Linux, Mac,... Gewoon even zoeken op de site en weg ben je.
- veel in- en uitgangen (analoge/ digitale, PWM) om een experimentje te wagen
- stroomvoorziening wordt automatisch gedetecteerd, via USB of externe voeding
- seriële uitlezing via de compiler: je kan op de PC volgen wat er gebeurt na een programmawijziging zonder dat de LCD-uitlezing aangesloten is.
- de 1280-chip (of hoger) heeft 2 uitgangen die speciaal gemaakt zijn om I2C-type LCD’s aan te sturen
- geïntegreerde polyfuse (zelfherstellende zekering) om de USB-poort (of zelfs het moederbord) van de PC te beveiligen ingeval van ‘ramp’.

In- en uitgangen van het Arduinobord

Het is de bedoeling om de sturing te kunnen gebruiken met allerlei soorten rotoren door het aansturen van 2 digitale uitgangen die een transistor of een optocoupler schakelen, die op zijn beurt een relais of ander element schakelt. Ook rotoren met een ‘rem’, zoals de HAM-4, komen in aanmerking. Voor de rem hoef je maar 1 lijntje bij te programmeren met een tijdsvertraging van bijvoorbeeld een halve seconde, uit- of inschakelen via een apart relais(tje). Dan heb je een extra digitale uitgang te programmeren, maar er zijn genoeg uitgangen.

Ik beveel je aan om bij alle experimenten een weerstand te plaatsen op de uitgang naar een LED of andere component, dit om ervoor te zorgen dat er geen uitgang QRT zou gaan of zelfs de chip in zijn geheel! De uitgangen zijn gelimiteerd tot 40 mA, maar een kortsluiting is vlug gemaakt. Ook is het beperken van het afgeleverd vermogen aan te bevelen (hoe minder, hoe beter).

Ik gebruik een voeding van een oude PC om mijn antennerotor (sturing met een ruitenwissermotor van een wagen, met reductiesysteem) van stroom te voorzien. Zorg ervoor dat alle massa’s, print en externe voeding, goed aan elkaar liggen. Dus de ‘ground’ van de print en externe voeding aan het chassis leggen. Er zijn meerdere ground-uitgangen op het bord. Speel op zeker en soldeer er een paar vast aan het chassis.

Twee analoge ingangen worden gebruikt voor het uitlezen van 2 potentiometers: 1 buiten aan de rotor en 1 binnen aan de sturing, om de richting te kiezen. Ik wou een volautomatische werking: als ik een draai geef aan de sturing naar eender welk werelddeel, moet de uitgang zodanig geset wordt dat die naar die richting gaat. Dus gebruiken we 2 analoge ingangen en 2 digitale uitgangen die via een transistor een relais schakelen, links of rechts. Met een rotorrem gebruik je 3 digitale uitgangen. Op het bord is standaard op elke digitale uitgang een SMD-LED gesoldeerd die aangeeft dat de schakeling in werking is (rotor aan het draaien). Zo kan je ook zonder rotor of relais nagaan of alles werkt, om zaken te testen als er iets misloopt of als je iets wilt wijzigen.

Mon premier programme

N.d.l.r. Le code source des programmes est affiché de manière abrégée à la fin de cet article. La version intégrale est disponible sur le site de l'auteur: users.telenet.be/ON4ADN/.

Le programme est écrit en C. Sur le forum d'Arduino, on peut trouver toutes les explications nécessaires pour démarrer. Dans le programme, tout ce qui est précédé d'un double slash (//) est considéré par le compilateur comme du commentaire. Ces commentaires sont utiles pour s'y retrouver par la suite dans le programme.

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD
address to 0x27 for a 16 chars and 2 line display
#include <Wire.h>
```

Dans l'en-tête du programme, on peut voir ce que j'utilise comme hardware: un afficheur LCD à deux fils via une liaison I2C. Toute l'information pour l'afficheur est transmise sur deux fils, mais il est possible de choisir un autre type d'afficheur. Il y a des afficheurs LCD très bon marché nécessitant plusieurs entrées ou sorties, mais rassurez-vous, il y en a en suffisance sur la carte.

Les bibliothèques de programme pour le hardware utilisé sont définies en début de programme.

```
const int sensorPin1= A0; // potentiometer 1
const int sensorPin2= A3; // potentiometer 2
```

Les entrées analogiques pour les deux potentiomètres sont appelées A0 et A3.

```
int diff = 2;
```

Cette instruction est utilisée pour compenser la différence de direction. Pratiquement toutes les antennes soumises au vent oscillent un peu autour de leur position, cela entraîne une variation constante de la valeur du potentiomètre de rotor. La puissance de calcul du 1280 est telle que les sorties de commande gauche ou droite seraient constamment activées. Le rotor n'y survivrait pas.

Pourquoi le chiffre 2? Cela représente la variation que l'on peut avoir sur le potentiomètre du rotor, sans qu'une sortie soit activée. Le microcontrôleur ne réagira pas dans cet intervalle. L'entrée analogique est alimentée en 5 VDC et la conversion A/D se fait avec une résolution de 10 bits. Le résultat est une valeur entière comprise entre 0 et 1023. Quel que soit l'affichage du LCD, le comptage de 0 à 1023 continue. Une rotation de 360° correspond à 1023, une rotation de 1° correspond donc pour l'entrée à une variation de 1023/360 = +/- 2,841666. L'instruction *int diff = 2* correspond donc à une variation de 2 degrés environ, 1 en plus ou 1 en moins par rapport à la valeur de consigne entrée; ceci afin d'éviter une commutation constante des sorties. Si on ne fixe pas cette valeur de *diff*, le risque est grand d'avoir une sortie gauche ou droite continuellement activée. En HF, l'effet de ce degré en plus ou en moins est négligeable, mais si une plus grande précision est désirée, on peut fixer la valeur de *diff* à 1 ou même 0,5. Cette valeur peut être adaptée suivant les besoins, juste un petit calcul à faire. Même la valeur zéro est possible si l'on est sûr que tout est bien fixe, mais ce n'est pas conseillé.

```
void setup()
```

Cette partie du programme n'est exécutée qu'une fois par le microcontrôleur. J'ai programmé ici sur le display, un petit message de bienvenue qui reste affiché 8 secondes après le démarrage:

```
 //zet begin op lijn 1 op afstand 1 en opgepast:
lijn 1 op de lcd = 0 daarom (1, 0)
  //waarbij de 1 zegt waar het eerste karakter
```

Mijn eerste programma

N.v.d.r. de broncode wordt verkort weergegeven achteraan dit artikel. De volledige versie kan je terugvinden op de website van de auteur: users.telenet.be/ON4ADN/.

Het programma is geschreven in de programmeertaal C. Op het Arduino-forum kan je uitgebreid terugvinden hoe je daarbij het best tewerkgaat. Alles wat na dubbele schuine streepjes (//) staat in het programma is 'commentaar', nuttig om later je weg terug te vinden in het programma. Deze commentaarlijnen worden uiteraard door de compiler overgeslagen.

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD
address to 0x27 for a 16 chars and 2 line display
#include <Wire.h>
```

In de header van het programma zie je wat ik gebruik als hardware: een tweedraads-LCD via een I2C- verbinding. Alle informatie naar het LCD gaat over 2 draden, maar de keuze van een ander LCD staat vrij. Er zijn erg goedkope LCD-schermen te verkrijgen waar je meerdere in- of uitgangen voor nodig hebt, maar wees gerust, je hebt er meer dan genoeg op het printje.

Aan het begin van het programma worden de programmabibliotheken voor de gebruikte hardware gedefinieerd.

```
const int sensorPin1= A0; // potentiometer 1
const int sensorPin2= A3; // potentiometer 2
```

Hier benoem ik de analoge ingangen voor de 2 potentiometers, A0 en A3.

```
int diff = 2;
```

Dit is een regel die gebruikt wordt om het richtingsverschil te compenseren. Bijna alle antennes staan in de wind een beetje te wiebelen en de potentiometer in de rotor zou iedere keer een andere waarde kunnen aannemen. De rekenkracht van de 1280 chip is zo groot dat hij iedere keer een uitgang links of rechts zou setten, continu. De rotor zou dit niet lang volhouden.

Waarom het cijfer 2? Dat is het verschil dat men mag hebben op de potentiometer in de rotor zonder dat een uitgang geset wordt (tolerantie). De microcontroller zal dan binnen die grenzen niet reageren. De analoge ingang wordt met 5 VDC gestuurd en leest in een 10-bit resolutie. De uitkomst is een integerwaarde van 0 tot 1023. Wat er ook op het LCD komt, de telling van 0-1023 blijft doorgaan. Het draaibereik is 360° van onze rotor op een telling van 0-1023 en dat maakt dus dat men bij iedere telling van 1° graad op +/- 2,841666 uitkomt (1023 gedeeld door 360°). Het *int diff = 2* bedraagt dus ongeveer 2 graden, 1 plus of 1 min de ingestelde waarde binnen aan de bediening, dit om de uitgangen niet constant te doen schakelen. Indien men deze waarde niet instelt, is de mogelijkheid groot dat er continu een uitgang links/rechts geset wordt. Op HF is die +/- 1 graad te verwaarlozen, maar als je het nauwkeuriger wilt, zet dan de waarde van *diff* gelijk aan 1 of zelfs 0,5. Die waarde mag naar believen aangepast worden, gewoon even rekenen en aanpassen. Zelfs een waarde 0 is mogelijk als men zeker is dat alles muurvast is, maar dat is niet aan te raden.

```
void setup()
```

Dit gedeelte van het programma wordt eenmaal uitgevoerd door de microcontroller. Hier heb ik op het display een welkomstboodschap geprogrammeerd die na de start 8 seconden aan blijft:

```
 //zet begin op lijn 1 op afstand 1 en opgepast:
lijn 1 op de lcd = 0 daarom (1, 0)
  //waarbij de 1 zegt waar het eerste karakter
```

```
moet staan en de 0 de bovenste lijn is
 lcd.setCursor(1, 0);
 //tekst voor lcd altijd tussen aanhalingstekens
plaatsen
 lcd.print("HELLO ON4ADN");
 //lijn 2 is de tweede 1 (1, 1)
 lcd.setCursor(1, 1);
 lcd.print ("WELCOME ON HF");
 //tijd om het welkomsscherm te zien, 1000 = 1
seconde
 delay (8000);
 // wis het scherm
 lcd.clear();
```

Le nombre 8000 correspond à 8 secondes et peut être adapté. Le texte qui doit apparaître sur l'afficheur se trouve toujours entre guillemets.

```
//set output pinnen 41 up, 29 down, 13 ok
 pinMode(41, OUTPUT); // motor links
 pinMode(29, OUTPUT); //motor rechts
 pinMode(13, OUTPUT); //motor stop
```

Ici, les sorties digitales sont nommées. La sortie 13 est une sortie avec LED SMD intégrée, de sorte que l'on peut voir si le rotor tourne, même en l'absence de celui-ci.

```
void loop()
```

A partir d'ici, le programme s'exécute de façon continue.

```
 int Val1 = analogRead(sensorPin1); // waarde
potentiometer 1
 int Val2 = analogRead(sensorPin2); // waarde
potentiometer 2
 //als potentiometer 2 kleiner is dan potentio-
meter1 - diff, 41 high en 13 high
 if (Val2<Val1-diff) {
 digitalWrite (41, HIGH);
 digitalWrite (29, LOW);
 digitalWrite (13, HIGH);
```

Ceci est un exemple de la lecture continue des potentiomètres. Les valeurs des deux potentiomètres sont mesurées de façon continue, de 0 à 1023. digitalWrite signifie qu'une sortie est activée ou non, LOW ou HIGH, et la sortie 13 est la LED intégrée. Donc, si la différence *(diff)* est plus grande ou plus petite que la valeur entrée, une sortie est activée ou non.

```
 if (Val2 >0 && Val2 <=2.8416)//set 180°
 {
 lcd.setCursor(9, 0);
 lcd.print("Ant=180 ");
 }
```

Ici, on peut voir comment je commande l'afficheur et lui fait afficher ce que je veux. Val2 (valeur du potentiomètre à l'antenne) est supérieur à zéro et inférieur ou égal à 2,8416. Encore une fois: nous calculons de 0 à 1023 sur 360° et disons que ce nombre ne doit être affiché que si cette valeur tombe dans l'intervalle. Pour 0 Ω, je fais afficher 180° et pour 10 kΩ, je fais aussi afficher 180°. Au milieu du potentiomètre, par exemple 5 kΩ pour un potentiomètre de 10 kΩ, je fais afficher 360° pour passer ensuite à 0° jusqu'au sud à 180°. Mais le comptage se fait toujours de 0 à 1023, **quelle que soit la valeur du potentiomètre**. Peu importe que l'on ait par exemple à l'extérieur un potentiomètre de 1 kΩ et à l'intérieur un potentiomètre 10 tours de 10 kΩ, le microcontrôleur compte toujours de 0 à 1023.

J'ai dû programmer chaque degré séparément. Je ne voyais pas d'autre solution mathématique que celle-là, pour passer au nord,

```
moet staan en de 0 de bovenste lijn is
 lcd.setCursor(1, 0);
 //tekst voor lcd altijd tussen aanhalingstekens
plaatsen
 lcd.print("HELLO ON4ADN");
 //lijn 2 is de tweede 1 (1, 1)
 lcd.setCursor(1, 1);
 lcd.print ("WELCOME ON HF");
 //tijd om het welkomsscherm te zien, 1000 = 1
seconde
 delay (8000);
 // wis het scherm
 lcd.clear();
```

Het getal 8000 is 8 seconden en kan aangepast worden. De tekst die moet verschijnen op het display staat steeds tussen aanhalingstekens.

```
//set output pinnen 41 up, 29 down, 13 ok
 pinMode(41, OUTPUT); // motor links
 pinMode(29, OUTPUT); //motor rechts
 pinMode(13, OUTPUT); //motor stop
```

Hier worden de digitale uitgangen benoemd. Uitgang 13 is er één met ingebouwde SMD LED, zodat je kan zien of de rotor draait, zelfs zonder dat de rotor aanwezig is.

```
void loop()
```

Vanaf hier wordt het programma continu doorlopen.

```
 int Val1 = analogRead(sensorPin1); // waarde
potentiometer 1
 int Val2 = analogRead(sensorPin2); // waarde
potentiometer 2
 //als potentiometer 2 kleiner is dan potentio-
meter1 - diff, 41 high en 13 high
 if (Val2<Val1-diff) {
 digitalWrite (41, HIGH);
 digitalWrite (29, LOW);
 digitalWrite (13, HIGH);
```

Dit is een voorbeeld van de uitlezing die continu wordt doorlopen. De waarden van de beide potentiometers worden continu gemeten, van 0 – 1023. digitalWrite wil zeggen dat een uitgang wel of niet gestuurd wordt, LOW of HIGH, en uitgang 13 is de inwendige LED. Dus als het verschil *(diff)* kleiner of groter is dan ingesteld, wordt een uitgang al dan niet geset.

```
 if (Val2 >0 && Val2 <=2.8416)//set 180°
 {
 lcd.setCursor(9, 0);
 lcd.print("Ant=180 ");
 }
```

Hier zie je hoe ik het display aanstuur en laat schrijven wat ik wil. Val2 (waarde potmeter aan antenne) is groter dan 0 en kleiner dan of gelijk aan 2,8416. Nogmaals: we rekenen van 0 – 1023 over 360° en zeggen alleen maar dat dit getal op het display moet komen als die waarde binnen het bereik valt. Bij 0 Ω laat ik 180° 'schrijven' en bij 10 kΩ laat ik ook 180° schrijven. In het midden van de pot, bijvoorbeeld 5 kΩ op een 10 kΩ potentiometer laat ik 360° schrijven om dan over te gaan naar 0° tot in het zuiden op 180°. Maar de telling blijft altijd lopen van 0 tot 1023, **ongeacht** de waarde van de potentiometer. Het mag buiten een potentiometer zijn van 1 kΩ en binnen een 10-slagen- pot van 10 kΩ, het doet er niet toe, de microcontroller telt immers van 0-1023.

Elke graad heb ik apart moeten programmeren. Ik kon maar niet bedenken hoe ik het wiskundig anders kon aanpakken om in het noor-

de 360° à 0°. Dans certains continents où le comptage se fait de 0 à 360°, la programmation est beaucoup plus simple.

Sur la première ligne de l'afficheur, je fais apparaître la valeur de consigne et la valeur mesurée. Dans l'instruction *lcd.setCursor* on voit 2 nombres entre guillemets. Nous avons un afficheur de 16 caractères par ligne et ici, l'instruction dit que 180 doit apparaître à la neuvième place sur la première ligne. Le chiffre 0 correspond à la première ligne et le chiffre 1 à la seconde, etc. Dans le cas d'un afficheur LCD à 4 lignes, les lignes sont désignées par les chiffres 0,1,2 ou 3. Le passage de 360° à 0° se fait lorsque le compteur atteint 512, pour les deux potentiomètres (Val1 en Val2), ensuite, de 0° on revient à 180°.

```
 // tussen 45 en 60°
if (Val2 >620 && Val2<=660)
{
 lcd.setCursor(0, 1);
 lcd.print("T8 UI8 VK9L BV ");
}
```

Je voulais d'abord utiliser les deux lignes de l'afficheur pour lire les valeurs des deux potentiomètres, mais c'était une erreur car chaque ligne comporte 16 caractères. J'ai donc placé les valeurs des deux potentiomètres sur la première ligne et en extra sur la seconde ligne, les préfixes des pays DXCC en fonction de la direction de l'antenne. J'ai fait une division de la surface terrestre de 0 à 1023. Si (cas de l'exemple) Val2 tombe entre 620 en 660, cela correspond à une direction d'antenne comprise entre 45° et 60°, et par conséquent aux préfixes T8 UI8 VK9L BV.

Mon deuxième programme

Cela m'ennuyait que mon premier programme prenait tant de place et j'ai recherché d'autres solutions. Après avoir lu sur le forum Arduino comment travailler avec des signaux PWM, j'ai poursuivi mes expérimentations dans ce sens. Les sorties PWM ne sont pas utilisées dans mon setup et c'est alors que j'ai compris comment on peut afficher n'importe quel nombre sur l'écran LCD via une capture sur la sortie analogique PWM. Ce deuxième programme occupe à peine 0,6 KB et peut être programmé pour n'importe quel chip Atmel. Texte et explications à volonté et cela peut même être programmé avec moins de 0,5 KB, tout dépend de ce que vous voulez voir apparaître sur l'écran LCD. Au dernier moment, j'ai alors encore programmé un texte visuel supplémentaire, pour voir dans quelle direction (environ) mon antenne pointe.

Il s'avère que cette version de programme est tellement sensible que le processeur produit de temps en temps du texte en double ou plus. Cela s'explique par le fait que le curseur du potentiomètre a une course trop grande pour la sortie PWM. Cette sortie "voit" alors plusieurs positions et veut les afficher sur le LCD. Ce problème est résolu en utilisant un potentiomètre multitours plus précis.

Compiler

L'utilisation des programmes écrits actuellement sous forme de texte n'est pas difficile.

- Démarrer le compilateur.
- Sélectionner (Ctrl-A) tout le texte (pour ne pas oublier des ";" ou des "{").
- Copier tout le texte.
- Tout coller (paste) dans le compilateur.

Le compilateur fait en sorte que tout vienne à sa place.

Matériel

J'utilise un rotor de 12 V, mais comme la commande se fait via deux relais, la tension d'alimentation (AC of DC) du rotor est sans importance. J'emploie un relais avec trois contacts inverseurs, dont un pour la sécurité, car un relais collant provoquerait un court-circuit. Donc: 1

den van 360° terug naar 0° gaan. In bepaalde werelddelen waar de telling gaat van 0 naar 360 loopt, is het programma veel eenvoudiger.

Op lijn 1 van het display laat ik beide waarden - de gevraagde waarde en de werkelijke waarde buiten - verschijnen. Bij *lcd.setCursor* zie je 2 getallen tussen de haakjes staan. We hebben een display van 16 tekens per lijn en hier zeg ik dat 180 op de negende plaats moet verschijnen op regel 1. Het cijfer 0 geeft regel 1, cijfer 1 regel 2, enz. Bij een 4-lijnen-LCD geeft 0,1,2 of 3 de regel aan. Op tellerstand 512 wordt de overgang gemaakt van 360° naar 0°, voor beide pots (Val1 en Val2), om dan terug van 0° naar 180° te gaan.

```
 // tussen 45 en 60°
if (Val2 >620 && Val2<=660)
{
 lcd.setCursor(0, 1);
 lcd.print("T8 UI8 VK9L BV ");
}
```

Ik zou eerst de twee lijnen van het LCD gebruiken om de waarde van de 2 pots uit te lezen, maar dat was zonde omdat elke lijn 16 tekens kon weergeven. Dus heb ik beide pots op de eerste lijn geplaatst en als extraatje op de tweede lijn: de DXCC-landen met hun prefix in de antennerichting. Ik heb een verdeling van de wereldbol gemaakt van 0-1023. Val2 ligt tussen de 620 en 660. Dit komt overeen met een antennerichting tussen 45° en 60°, hetgeen op zijn beurt overeenstemt met prefixen T8 UI8 VK9L BV.

Mijn tweede programma

Ik had het er moeilijk mee dat mijn eerste programma zoveel plaats in beslag nam en ging op zoek naar andere oplossingen. Op het Arduino-forum las ik hoe je PWM-signalen kon inlezen en experimenteerde daarmee verder. De PWM-uitgangen worden in mijn setup niet gebruikt en toen viel mijn euro hoe je eender welk getal kon plaatsen op de LCD via een uitlezing op de analoge PWM-uitgang. Dit tweede programma neemt amper 0,6 KB in beslag en kan voor eender welke Atmel-chip geprogrammeerd worden. Tekst en uitleg is naar believen en het kan zelfs met minder dan 0,5 KB geprogrammeerd worden, louter een kwestie van hoe je het wil zien op de LCD. Op het laatste moment heb ik dan nog een extra visuele tekst geprogrammeerd om te zien in welke richting mijn antenne (ongeveer) staat.

De verkorte versie is zo gevoelig dat de controller soms dubbele of driedubbele tekst weergeeft. Dit komt omdat de loper van de potentiometer een te groot bereik heeft voor de PWM-uitgang en meerdere standen 'ziet' en die meerdere standen op het LCD wil zetten. Met een preciezere, multiturnpotentiometer is dit probleem van de baan.

Compileren

Het gebruik van de geschreven programma's die nu in tekstformaat staan, is niet moeilijk.

- start de compiler op
- "alles selecteren" (Ctrl-A) van het tekstbestand (om geen ";" of "{" te vergeten)
- alles "kopiëren" van het tekstbestand
- in de compiler alles "plakken" (paste)

De compiler zorgt zelf wel dat alles op zijn plaats komt.

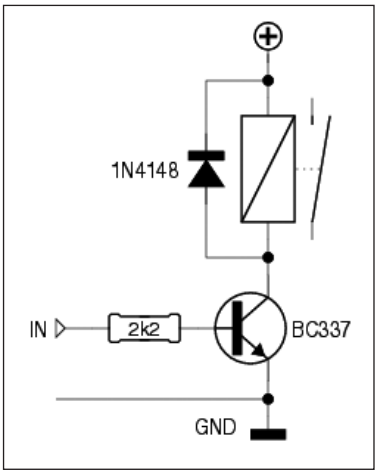
Hardware

Ik gebruik een 12 V rotor, maar vermits de sturing gebeurt via twee relais, speelt de rotorspanning geen rol (AC of DC). Ik gebruik een relais met 3 wisselcontacten, waarvan 1 contact als veiligheid, want een 'plakkend' relais zou een kortsluiting veroorzaken. Dus: 1 uitgang (van de rechtse

sortie (de la commande “droite”) est raccordée à un contact du relais de la commande “gauche”, l'autre contact via l'autre relais, et ainsi, si un relais ne revient pas au repos, l'autre ne peut pas être activé.

La commande se fait de préférence via un transistor ou un optocoupleur, ceci dans le but de protéger les sorties. La consommation de courant des sorties diminuera alors drastiquement. Des relais de 5 VDC conviennent très bien aussi, mais alors de préférence avec une consommation de bobine aussi basse que possible. N'oubliez pas de placer (dans le bon sens) une diode en parallèle sur la bobine du relais, sans quoi, le transistor risque de passe très vite en QRT.

Encore un truc: vérifiez via le port série (écran pc) que le potentiomètre couplé au rotor va bien de 0 à 1023 (voir Ham 4 ci-dessous). Les valeurs peuvent être adaptées dans le programme. Supposons que celui-ci commence à 2 (suite à une imperfection mécanique), alors, ce nombre peut être adapté de façon simple. Le plus souvent, pour corriger ces imperfections lors de lectures analogiques, les capteurs analogiques sont adaptés au moyen de résistances. J'ai programmé ici les nombres de façon un peu trop précise, ce qui n'était pas nécessaire, certainement pas pour des antennes HF. Mais c'est maintenant comme ça.



Autre truc pour un rotor Ham 4 utilisant un potentiomètre avec chevauchement au début et à la fin (**voir photo**). Raccordez le potentiomètre de rotor à une entrée analogique et contrôlez visuellement le début et la fin du “sud”. Cela commencera probablement aux environs de 30 pour terminer aux environs de 1000. Donc, il vous faudra adapter les nombres au début et à la fin dans le programme (j'emploie un potentiomètre 360° et donc pour moi, un tour complet va de 0 à 1023). Vous pouvez alors réaliser un chevauchement d'une dizaine de degrés au sud, au début et à la fin. Par mesure de sécurité, laissez clignoter l'affichage, de façon à voir que vous passez au-dessus de 180°. A vous de voir quel message vous voulez voir apparaître. L'adaptation des nombres peut prendre un certain temps.

```
void setup() {
Serial.begin(9600);
}
void loop() {
int sensorValue = analogRead(A0);
Serial.println(sensorValue);
}
```

Le code ci-dessus permet de visualiser sur l'écran de votre PC les valeurs acquises sur l'entrée analogique 0.

En guise de conclusion

Le microcontrôleur ATmega est puissant et constitue une source inépuisable pour toutes sortes de projets. Plusieurs sorties sont prévues, par exemple PWM (Pulse-width Modulation) pour piloter des moteurs pas à pas. Un exemple parmi d'autres est le pilotage d'une antenne cadre multibande. C'est une petite machine très précise pour effectuer des mesures et des commandes dans le cadre de notre hobby.

Je ne suis pas un programmeur et ceci est mon premier travail en C, mais j'ai quand même pu le mener à bonne fin. Si vous êtes bloqués quelque part, il y a sûrement quelqu'un dans votre club UBA qui pourra vous aider. Parfois, un petit tuyau a plus de valeur que tout un livre. J'ai pris goût à la chose et il y a encore deux cartes en commande pour deux projets supplémentaires, comme la commande de ventilateurs PA via une mesure de la température avec un capteur de température LM35DZ.

Plus je travaille avec ce microcontrôleur et plus je suis convaincu que le projet Arduino constitue vraiment un plus pour notre hobby. Vous pouvez

uitsturing) gaat over een contact van het relais dat links moet draaien, het andere over het andere relais, en indien niet in ‘rust’ kan het andere relais nooit aantrekken.

Het aansturen gebeurt het best via een transistor of optocoupler, dit om de uitgangen te beschermen. Het stroomverbruik zal drastisch verlagen, maar 5 VDC relais werken ook prima, maar dan best met een zo laag mogelijk verbruik van de spoel. Vergeet de diode over de spoel niet, of de transistor is geen lang leven beschoren.

Nog een tip: ga via de seriële poort (scherm pc) na of de potentiometer buiten aan de rotor wel tot 0 en 1023 loopt (zie Ham 4 hieronder). Je kan de waarden in het programma aanpassen. Stel dat die maar (door mechanische onvolmaaktheden) begint aan 2, dan kan je dat getal eenvoudig aanpassen. Meestal worden analoge meters aangepast met 2 weerstanden om die onvolmaaktheid bij analoge meting tegen te gaan. Ik heb de getallen hier wel een beetje te precies geprogrammeerd en dat hoefde niet, zeker niet voor HF-antennes. Maar het staat er nu eenmaal.

Nog een tip, bijvoorbeeld voor een Ham4-rotor waar gebruik gemaakt wordt van een potentiometer die aan het begin en het einde een soort overlapping heeft (**zie foto**). Koppel de rotorpotentiometer aan een analoge ingang, ga na waar het zuiden begint en eindigt (visueel). Dat zal waarschijnlijk rond de 30 beginnen en eindigen op een 1000. Dus zal je de getallen in het programma moeten aanpassen in het begin en op het einde (ik gebruik een 360° pot en bij mij is dat dus compleet rond van 0–1023). Dan kan je aan het begin en einde een overlapping maken op het zuiden van een tiental graden. Je kan dan veilgheidshalve de LCD laten knipperen, zodat je ziet dat je over de 180° bent. Wat je daarbij naar het LCD schrijft, laat ik aan je verbeelding over. De getallen aanpassen zal wel wat tijd vragen.

Met deze code kan je op het scherm van je PC de waarden zien die uitgelezen worden op de analoge ingang 0:

```
void setup() {
Serial.begin(9600);
}
void loop() {
int sensorValue = analogRead(A0);
Serial.println(sensorValue);
}
```

Tot slot

De ATmega microcontroller is krachtig spul en een onuitputtelijke bron voor allerlei projecten. Er zijn meerdere uitgangen voorzien, bijvoorbeeld PWM (Pulsewidth Modulation) om stappenmotoren te sturen. Denk bijvoorbeeld maar eens aan het sturen van een loopantenne voor meerdere banden. Dit bordje is een héél precies ding voor metingen en sturingen voor onze hobby.

Ik ben geen programmeur en dit is mijn eerste werk in C, maar ik heb dit toch tot een goed einde kunnen brengen. Indien je ergens vastloopt, kan iemand in de UBA-club je zeker verder helpen met info. Soms is een tip is meer waard dan een gans boek. Ik heb de goesting te pakken en er zijn nog 2 kleinere bordjes besteld met 2 extra projecten in het achterhoofd, zoals het sturen van PA-ventilatoren via een gemeten waarde met een LM35DZ temperatuur sensor.

Hoe langer ik ermee bezig ben, hoe meer ik overtuigd ben dat het Arduino-project voor onze hobby een echte meerwaarde is. Je kan er

beaucoup en apprendre et, avec un minimum de hardware, réaliser un beau projet. Les chips Atmel sont très bon marché, mais surtout, on peut les trouver sans le moindre problème sur internet. Sur la plateforme Arduino, on peut trouver tous les schémas pour réaliser des prints avec un chip Atmel, ce qui est un très bon point. Il y a des exemples à profusion, il est aussi possible de tester le compilateur en ligne.

Attention, ainsi que quelqu'un me l'a un jour écrit: une fois que vous commencez avec ça, c'est comme une drogue et vous ne pouvez plus vous en passer!

Pour de plus amples informations, envoyez-moi un e-mail.

73 Geert
Geert.Decru@Telenet.be

Bibliographie:
http://www.arduino.cc/
http://www.arduino.nu/

veel van opsteken en met minimale hardware iets in elkaar boksen. De Atmel-chips zijn zeer goedkoop te verkrijgen, maar vooral zonder enig probleem te vinden op het internet. Alle schema's om een print te maken met een Atmel-chip staan op het Arduino platform, dus dat is ook al zeer positief. Voorbeelden staan er genoeg, ook in de compiler kan je die oproepen en eens uitproberen.

Opgepast, zoals er mij ooit iemand schreef: als je daaraan begint is het als een drug, het werkt verslavend!

Verdere info nodig? Graag, via e-mail.

73 Geert
Geert.Decru@Telenet.be

Bibliografie:
http://www.arduino.cc/
http://www.arduino.nu/

Programme 1 / Programma 1

```
//Created on 3/2012 for HAM radio use as an universal digital antenna control box
//This software can work with all rotor controllers with potentiometer inside the rotor
// for info: you can find me on the internet, 73's Geert
// language is in Dutch but if you need I can translate this in English

// ON4ADN / OO2A


#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD address to 0x27 for a 16 chars and 2 line display

#include <Wire.h>

const int sensorPin1= A0; // potentiometer 1
const int sensorPin2= A3; // potentiometer 2
int diff = 2; //verschil om niet altijd links of rechts uit te sturen
 //gezien de potentiometer buiten niet perfect constant blijft zou die altijd links of rechts uitsturen
 //in te stellen volgens goesting
 //ingebouwde LED 13 is voor het testen van de sturing op tafel, anders mag dat gewist worden
 //LED 13 mag ook een andere uitgang zijn en een LED aansturen op de bediening om te zien of het OK is
 //Outputs 41 en 29 zijn willekeurig gekozen

 void setup() {
 lcd.init(); // initialize the lcd

 // Print een boodschap op de LCD.
 //lcdbacklight is een lcd1602 I2C met achtergrond verlichting
 lcd.backlight();
 //zet begin op lijn 1 op afstand 1 en opgepast: lijn 1 op de lcd = 0 daarom (1, 0)
 //waarbij de 1 zegt waar het eerste karakter moet staan en de 0 de bovenste lijn is
 lcd.setCursor(1, 0);
 //tekst voor lcd altijd tussen aanhalingstekens plaatsen
 lcd.print("HELLO ON4ADN");
 //lijn 2 is de tweede 1 (1, 1)
 lcd.setCursor(1, 1);
 lcd.print ("WELCOME ON HF");
```

Programme 2 / Programma 2

```
// Created 1/2012
// by Geert Decru ON4ADN


#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD address to 0x27 for a 16 chars and 2 line display

#include <Wire.h>

const int sensorPin1= A0; // potentiometer 1
const int sensorPin2= A3; // potentiometer 2
const int analogOutPin2 = A2; // Analog output pin
const int analogOutPin3 = A3; // Analog output pin
int diff = 2; //verschil om niet altijd links of rechts uit te sturen
 //gezien de potentiometer buiten niet perfect constant blijft zou die altijd links of rechts uitsturen
 //in te stellen volgens goesting
 //ingebouwde LED 13 is voor het testen van de sturing op tafel, anders mag dat gewist worden
 //LED 13 mag ook een andere uitgang zijn en een LED aansturen op de bediening om te zien of het OK is
 //Outputs 41 en 29 zijn willekeurig gekozen
 int outputValue2 = A2; // value output to the PWM (analog out)
 int outputValue3 = A3; // value output to the PWM (analog out)

 void setup() {
 lcd.init(); // initialize the lcd

 // Print een boodschap op de LCD.
 //lcdbacklight is een lcd1602 I2C met achtergrond verlichting
 lcd.backlight();
 //zet begin op lijn 1 op afstand 1 en opgepast: lijn 1 op de lcd = 0 daarom (1, 0)
 //waarbij de 1 zegt waar het eerste karakter moet staan en de 0 de bovenste lijn is
 lcd.setCursor(1, 0);
 //tekst voor lcd altijd tussen aanhalingstekens plaatsen
 lcd.print("HELLO ON4ADN");
 //lijn 2 is de tweede 1 (1, 1)
 lcd.setCursor(1, 1);
 lcd.print ("WELCOME ON HF");
 //tijd om het welkomsscherm te zien, 1000 = 1 seconde
```

Programme 1 (suite) / Programma 1 (vervolg)

```
 //tijd om het welkomsscherm te zien, 1000 = 1
seconde
 delay (8000);
 // wis het scherm
 lcd.clear();

 //set output pinnen 41 up, 29 down, 13 ok
 pinMode(41, OUTPUT); // motor links
 pinMode(29, OUTPUT); //motor rechts
 pinMode(13, OUTPUT); //motor stop
 }

 void loop() {
  int Val1 = analogRead(sensorPin1); // waarde
potentiometer 1
  int Val2 = analogRead(sensorPin2); // waarde
potentiometer 2
 //als potentiometer 2 kleiner is dan potentio-
meter1 - diff, 41 high en 13 high
 if (Val2<Val1-diff) {
 digitalWrite (41, HIGH);
 digitalWrite (29, LOW);
 digitalWrite (13, HIGH);
 }

 //als potentiometer 2 groter is dan potentio-
meter 1 + diff, 29 high en 13 high
 else if (Val2>Val1+diff) {
 digitalWrite (41, LOW);
 digitalWrite (29, HIGH);
 digitalWrite (13, HIGH);
 lcd.setCursor(16, 0);
 lcd.print("T");
 }

 // als alles binnen de grenzen van verschil van
2 graden is alle uitgangen low
 else {
 digitalWrite (41, LOW);
 digitalWrite (29, LOW);
 digitalWrite (13, LOW);

 }

 if (Val2 >0 && Val2 <=2.8416)//set 180°
 {
 lcd.setCursor(9, 0);
 lcd.print("Ant=180 ");
 }
 if (Val2 >2.8416 && Val2<=5.6832)//set 181°
 {
 lcd.setCursor(9, 0);
 lcd.print("Ant=181 ");
 }

... enzoverder, voor elke graad...

 if (Val1 >1019 && Val1<=1020)
 {
 lcd.setCursor(0, 0);
 lcd.print("Goto:179 ");
 }
 if (Val1 >1020 && Val1<=1022.976)
 {
 lcd.setCursor(0, 0);
 lcd.print("Goto:180 ");
 }

 // tussen 1 en 15°
if (Val2 >500 && Val2<=540)
{
 lcd.setCursor(0, 1);
 lcd.print("3D2 ZL8 KH4 T30 ");
}
```

Programme 2 (suite) / Programma 2 (vervolg)

```
 delay (5000);
 // wis het scherm
 lcd.clear();

 //set output pinnen 41 up, 29 down, 13 ok
 pinMode(41, OUTPUT); // motor links
 pinMode(29, OUTPUT); //motor rechts
 pinMode(13, OUTPUT); //motor stop
 }

 void loop()
 {
  int Val1 = analogRead(sensorPin1); // waarde
potentiometer 1
  int Val2 = analogRead(sensorPin2); // waarde
potentiometer 2
  int sensorValue2 = analogRead(sensorPin1);//
sensor PWM1
  int sensorValue3 = analogRead(sensorPin2);//
output PWM
 int outputValue2 = analogRead(outputValue2);//
sensor PWM2
 int outputValue3 = analogRead(outputValue3);//
output PWM2


 //als potentiometer 2 kleiner is dan potentio-
meter1 - diff, 41 high en 13 high
 if (Val2<Val1-diff)
 {
 digitalWrite (41, HIGH);
 digitalWrite (29, LOW);
 digitalWrite (13, HIGH);
 delay(50);
 lcd.setCursor(15, 0);
 lcd.blink();
 delay(300);


 }

 //als potentiometer 2 groter is dan potentio-
meter 1 + diff, 29 high en 13 high
 else if (Val2>Val1+diff) {
 digitalWrite (41, LOW);
 digitalWrite (29, HIGH);
 digitalWrite (13, HIGH);
 delay(50);
 lcd.setCursor(15, 0);
 lcd.blink();
 delay(300);

 }

 // als alles binnen de grenzen van verschil van
2 graden is alle uitgangen low
 else
 {
 digitalWrite (41, LOW);
 digitalWrite (29, LOW);
 digitalWrite (13, LOW);
 lcd.noBlink();
 }

 if (Val1>=0 && Val1<=512)
 {
 sensorValue2 = analogRead(sensorPin1);
  //inlezing en uitlezing van een PWM signaal
integer waarde 0 tot 512 tot midden pot
 //en omgezet van 180 naar 360°
 outputValue2 = map(sensorValue2, 0, 512, 180,
360);
 analogWrite(analogOutPin2, outputValue2);
 lcd.setCursor(0, 0);
```

Programme 1 (suite) / Programma 1 (vervolg)

```
//tussen 15 en 30°
if (Val2 >540 && Val2<=580)
{
 lcd.setCursor(0, 1);
 lcd.print("C2 H40 T33 KH9 ");
}
//tussen 30 en 45°
if (Val2 >580 && Val2<=620)
{
 lcd.setCursor(0, 1);
 lcd.print("P5 FK JD1 VK9N ");
}
// tussen 45 en 60°
if (Val2 >620 && Val2<=660)
{
 lcd.setCursor(0, 1);
 lcd.print("T8 UI8 VK9L BV ");
}
//tussen 60 en 75°
if (Val2 >660 && Val2<=700)
{
 lcd.setCursor(0, 1);
 lcd.print("VK VR 1S0 3W BS7 ");
}
//tussen 75 en 90
if (Val2 >700 && Val2<=740)
{
 lcd.setCursor(0, 1);
 lcd.print("9N A5 VK9X VU4 ");
}
 // tussen 90 en 105°
if (Val2 >740 && Val2<=780)
{
 lcd.setCursor(0, 1);
 lcd.print("8Q VU7 ZL9 A4 EP ");
}
//tussen 105 en 120°
if (Val2 >780 && Val2<=820)
{
 lcd.setCursor(0, 1);
 lcd.print("E4 HZ VQ9 A7 YK ");
}

//tussen 120 en 135°
if (Val2 >820 && Val2<=860)
{
 lcd.setCursor(0, 1);
 lcd.print("3B6 7O FR FT/Z ");
}
//tussen 135 en 150°
if (Val2 >860 && Val2<=900)
{
 lcd.setCursor(0, 1);
 lcd.print("7Q 5X FT/W FT/X ");
}
 // tussen 150 en 165°
if (Val2 >900 && Val2<=960)
{
 lcd.setCursor(0, 1);
 lcd.print("3DA 5A C9 ZS8 4U ");
}

//tussen 165 en 180°
if (Val2 >960 && Val2<=1023)
{
 lcd.setCursor(0, 1);
 lcd.print("3C0 9Q TN V5 D2 ");
}
//tussen 180 en 195°
if (Val2 >0 && Val2<=40)
{
 lcd.setCursor(0, 1);
 lcd.print("3Y 5U ZD7 ZD9 TY ");
}
```

Programme 2 (suite) / Programma 2 (vervolg)

```
 lcd.print("Gevraagd = ");
 lcd.print(outputValue2);

 }
 if (Val1 >=513 && Val1 <=1023)
 {
 sensorValue2 = analogRead(sensorPin1);
 outputValue2 = map(sensorValue2, 513, 1023, 0,
180);
 //inlezing en uitlezing van over de helft van
de pot integer waarde 512 naar einde 1023
 //boven overgaan naar het getal 0 om te stop-
pen op 180°
 analogWrite(analogOutPin2, outputValue2);
 lcd.setCursor(0, 0);
 lcd.print("Gevraagd = ");
 lcd.print(outputValue2);
 lcd.setCursor(8, 1);
 }

 if (Val2 >=0 && Val2 <=512)
 {
 sensorValue3 = analogRead(sensorPin2);
 outputValue3 = map(sensorValue3, 0, 512, 180,
360);
 //inlezing en uitlezing van een PWM signaal
integer waarde 0 tot 512 tot midden pot
 //en omgezet van begin 180 naar 360 midden pot°
 analogWrite(analogOutPin3, outputValue3);
 lcd.setCursor(0, 1);
 lcd.print("Ant= ");
 lcd.print(outputValue3);
 }

 if (Val2 >=513 && Val2 <=1023)
 {
 sensorValue3 = analogRead(sensorPin2);
 outputValue3 = map(sensorValue3, 513, 1023, 0,
180);
 //inlezing en uitlezing van over de helft van
de pot integer waarde 512 naar einde 1023
 //boven overgaan van 360° naar het getal 0 om
te stoppen op 180°
 analogWrite(analogOutPin3, outputValue3);
 lcd.setCursor(0, 1);
 lcd.print("Ant= ");
 lcd.print(outputValue3);

 }
 if (Val2 >=0 && Val2 <=60)
 {
 lcd.setCursor(9, 1);
 lcd.print("Zuiden ");
 }
 if (Val2 >=61 && Val2 <=180)
 {
 lcd.setCursor(9, 1);
 lcd.print("ZuidWest ");
 }
 if (Val2 >=181 && Val2 <=315)
 {
 lcd.setCursor(9, 1);
 lcd.print("Westen ");
 }
 if (Val2 >=316 && Val2 <=440)
 {
 lcd.setCursor(9, 1);
 lcd.print("NooWest ");
 }
 if (Val2 >=441 && Val2 <=570)
 {
 lcd.setCursor(9, 1);
 lcd.print("Noorden ");
 }
```

Programme 1 (suite) / Programma 1 (vervolg)

```
//tussen 195 en 210°
if (Val2 >40 && Val2<=80)
{
 lcd.setCursor(0, 1);
 lcd.print("VP8 4K1 3X ZD8 ");
}

//tussen 210 en 225°
if (Val2 >80 && Val2<=120)
{
 lcd.setCursor(0, 1);
 lcd.print("3Y D4 PY0 VP8 EA8 ");
}
//tussen 225 en 240°
if (Val2 >120 && Val2<=160)
{
 lcd.setCursor(0, 1);
 lcd.print("CE CX ZP PY CE0 ");
}
 // tussen 240 en 255°
if (Val2 >160 && Val2<=200)
{
 lcd.setCursor(0, 1);
 lcd.print("CE0 GJ PZ FY 8R ");
}

//tussen 255 en 270°
if (Val2 >200 && Val2<=240)
{
 lcd.setCursor(0, 1);
 lcd.print("8P CE0Y VP2 KP4 ");
}

//tussen 270 en 285°
if (Val2 >240 && Val2<=280)
{
 lcd.setCursor(0, 1);
 lcd.print("6Y HC8 HK0 VP6 ");
}
//tussen 285 en 300°
if (Val2 >280 && Val2<=320)
{
 lcd.setCursor(0, 1);
 lcd.print("CY FO VP6 4U VE ");
}
//tussen 300 en 315°
if (Val2 >320 && Val2<=360)
{
 lcd.setCursor(0, 1);
 lcd.print("FO K OX GD XF4 GI ");
}
 // tussen 315 en 330°
if (Val2 >360 && Val2<=400)
{
 lcd.setCursor(0, 1);
 lcd.print("ES FO GM TF VE K6 ");
}

//tussen 330 en 345°
if (Val2 >440 && Val2<=470)
{
 lcd.setCursor(0, 1);
 lcd.print("KH5 KH5K KH7 T32 ");
}
//tussen 345 en 360°
if (Val2 >470 && Val2<=500)
{
 lcd.setCursor(0, 1);
 lcd.print("ZL7 ZK2 T31 5W ");
}

}
```

Programme 2 (suite) / Programma 2 (vervolg)

```
 if (Val2 >=571 && Val2 <=700)
 {
 lcd.setCursor(9, 1);
 lcd.print("NooOost ");
 }
 if (Val2 >=701 && Val2 <=830)
 {
 lcd.setCursor(9, 1);
 lcd.print("Oosten ");
 }
 if (Val2 >=831 && Val2 <=960)
 {
 lcd.setCursor(9, 1);
 lcd.print("ZuidOost ");
 }
 if (Val2 >=961 && Val2 <=1023)
 {
 lcd.setCursor(9, 1);
 lcd.print("Zuiden ");
 }

}
```