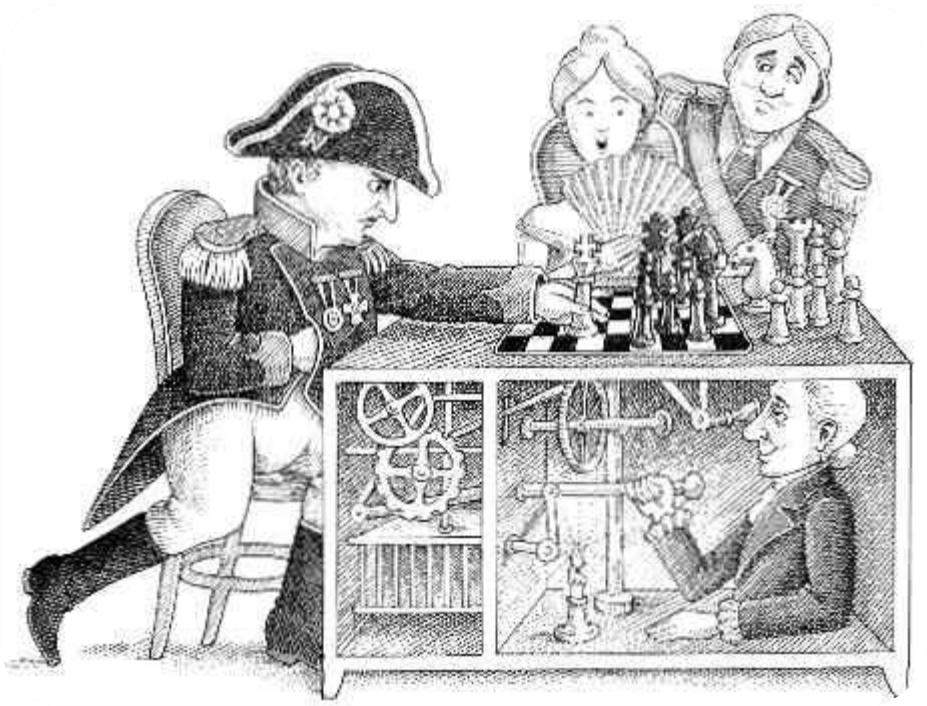
# Embedded systemen en kunstmatige intelligentie

Essay



**W. van Bakel**
**T. Weijers**

# Embedded systemen en kunstmatige intelligentie

Essay


Auteurs:

Wouter van Bakel 08014469
Tjeerd Weijers 08021198


Versie:

v1.0

Datum:

17 december 2010


Instelling:

TIS Delft – De Haagse Hogeschool

# Inhoudsopgave

# Verklarende woordenlijst

| | |
|---|---|
| KI | Kunstmatige Intelligentie |
| AI | Artificial Intellingence |
| Embedded systeem | Elektronisch systeem (hardware én software) dat is geïntegreerd in apparaten, met de bedoeling deze een vorm van intelligent gedrag te bezorgen |
| MIT | Massachusetts Institute of Technology |
| DARPA | Defense Advanced Research Projects Agency |
| HP | Hewlett-Packard |

# 1. Inleiding

Veel Embedded systemen hebben nog geen enkele vorm van kunstmatige intelligentie(KI) aan boord. Tegenwoordig worden Embedded systemen steeds sneller, bezitten meer geheugen en zijn energiezuiniger. Deze eigenschappen maken het mogelijk om kunstmatige intelligentie aan dergelijke systemen toe te voegen. Dit essay gaat dieper in op wat kunstmatige intelligentie eigenlijk inhoudt en hoe dat in ons hedendaags leven al is verwerkt. Ook wordt er proefondervindelijk gekeken of zeer kleine Embedded systemen ook al kunnen worden uitgerust met vormen van intelligentie.

De hoofdvraag van dit essay is: Wat is de toekomst van kunstmatige intelligentie in Embedded systemen?
Om antwoord op deze vraag te krijgen zijn er in dit essay een aantal deelvragen opgesteld:
Welke vormen van KI zijn er (KI, autolearning, neural netwerk)?
Zijn hedendaagse Embedded systemen geschikt voor kunstmatige intelligentie?
Hoe wordt KI in de huidige systemen toegepast?

# 2. Kunstmatige intelligentie

Kunstmatige intelligentie of in het Engels Artificial intelligence is wetenschap die zich bezighoudt met het creëren van een Artefact. Artefact komt uit het Latijn en betekend kunstmatig gemaakt.
Het is echter moeilijk te definiëren wat intelligentie precies is. Het is daarom ook moeilijk te definiëren wat artificiële of kunstmatige intelligentie precies is.[1]

David Wechsler, de Amerikaanse psycholoog die de beroemde Wechsler intelligentietests ontwierp definieerde intelligentie eenvoudig: *Intelligentie is het vermogen doelgericht te handelen, rationeel te denken en effectief met de omgeving om te gaan*.[2]

Toch moet er een manier zijn om te bepalen of een machine intelligent is. Deze manier bestaat in de vorm van een algemeen geaccepteerde test voor kunstmatige intelligentie; de Turing-test.
De Turing-test toont aan dat als een computer iemand voor de gek kan houden, en deze doen laten geloven dat hij een mens is, de computer intelligent moet zijn. Voor een dergelijke test moeten dan de omstandigheden zodanig worden gemaakt dat de proefpersoon niet ziet met wie hij praat, bijvoorbeeld door middel van chatten. Als de ondervrager niet consistent kan vertellen wie mens en wie machine is, doorstaat de machine de test. Dat is tot op heden nog geen enkele machine gelukt.[3]

Kunstmatige intelligentie is op te delen in twee stromen namelijk:

- Sterke KI
- Zwakke (toegepaste) KI

Sterke KI is onlosmakelijk verbonden met de mens. Sterke KI is een model van een mens. Dit model doorloopt dezelfde levensfasen als een mens. Als pasgeborene leert het taal en kennis van de wereld ook heeft het organen in eigen beheer. Uiteindelijk heeft het model een volwaardig mensenbrein ontwikkeld.
Edward Fredkin, van het MIT AI Lab stelt dat dergelijke sterke KI modellen in de toekomst mogelijk het mensenras als huisdier kunnen houden.[4] Andere wetenschappers stellen dat sterke KI helemaal nooit bereikt wordt.

Onder zwakke KI vallen de zogenoemde autolearning systemen en andere intelligente software. Een voorbeeld van zwakke KI wat al veelvuldig wordt toegepast in de huidige maatschappij is gezichtsherkenning.

In 1906 krijgt Cajal de Nobelprijs voor zijn ontdekking dat onze hersenen bestaan uit neuronen die door middel van dendrieten met elkaar verbonden zijn. Pas in 1949 ontdekt Hebb op welke manier mensen en dieren nieuwe dingen kunnen leren. Dit gebeurt ten eerste door nieuwe dendrieten te laten groeien die voor nieuwe verbindingen zorgen. Dit gebeurt voornamelijk in de eerste levensjaren. Daarna bestaat het leren voornamelijk uit het versterken of verzwakken van de verbindingen die door de dendrieten gemaakt worden.[5]

Uit een recent artikel blijkt dat DARPA en HP al ver gevorderd zijn als het gaat om het maken van kunstmatige intelligentie. HP heeft een chip gemaakt die gebruik maakt van memristors.
Een memristor is een uitbreiding op de meest bekende passieve componenten (spoel, weerstand en een condensator). Een memristor kan verklaard worden aan de hand van een waterleiding. Het water staat gelijk aan de lading. De weerstand van de memristor is de diameter van de pijp, hoe nauwer de pijp hoe hoger de weerstand. Weerstanden hebben in principe een vaste weerstand zoals pijpen een vaste diameter hebben. Een memristor is een waterleiding die zijn diameter kan aanpassen aan de hoeveelheid en de richting van het water. Als het water door de pijp stroomt in één richting wordt de diameter groter. Als het water in de andere richting stroomt wordt de diameter kleiner. Als het water niet meer stroomt bevriest de pijp diameter totdat het water weer gaat stromen.[6] Deze eigenschappen zijn ideaal bij het creëren van een geheugen element, vandaar de naam memristor waar mem voor memory staat.

De chip van HP lijkt op onze hersenen. De neuronen zijn geïmplementeerd in de vorm van transistoren, de dendrieten in de vorm van memristors en de connecties daartussen zijn gemaakt met nanowires.[7] Natuurlijk is er zoveel meer over de memristor te vertellen maar dat valt buiten het onderwerp van dit essay.

Kunstmatige neurale netwerken zijn een poging om de structuur van onze hersenen in een programma vast te leggen. Tot op heden vallen kunstmatige neurale netwerken nog onder zwakke KI toch heeft het de potentie om te groeien tot sterke KI omdat het een model is van onze hersenen. Met dit hoofdstuk is één van de deelvragen van dit essay beantwoordt, KI is namelijk opgedeeld in twee hoofdstromen. De huidige vormen van KI zijn allemaal nog zwakke KI.

# 3. Kunstmatige intelligentie in Embedded systemen

Dit hoofdstuk gaat in op KI in Embedded systemen. In de eerste paragraaf wordt besproken wat een intelligent Embedded systeem inhoudt. Later worden de voor- en nadelen opgesomd en worden er een aantal relevante voorbeelden van KI in Embedded systemen gegeven.

## 3.1. Wat is een intelligent Embedded systeem

Een intelligent systeem, is een systeem dat nieuwe problemen kan oplossen, problemen die niet zijn geconditioneerd in de soft- of hardware. Alvorens er dieper wordt ingegaan op KI in Embedded systemen worden er wat misverstanden opgehelderd:[8]

- Een systeem dat door een intelligent persoon is gemaakt is niet per definitie een intelligent systeem. Dit systeem kan dan wel slim lijken, maar kan slecht omgaan met veranderingen in de problemen die hij afhandelt
- Intelligente systemen zijn niet altijd beter dan normale systemen. Een racefiets kan gezien worden als een “dom” systeem daar het ontworpen is voor één functie: vlakke wegen. Onze benen zijn dan een intelligent systeem die passen zichzelf aan op de omgeving, over een heuvel, vlakke weg of door het water. In sommige gevallen is het toch verstandiger om de racefiets te kiezen
- Intelligente systemen zijn niet altijd complex, in het praktijkonderzoek bij dit essay wordt een voorbeeld gegeven van een simpel intelligent Embedded systeem

## 3.2. Voor- en nadelen van een KI Embedded systeem

Intelligente systemen kunnen op diverse gebieden voor- en nadelen bieden ten opzichte van conventionele systemen.[8]

**Aanpasbaarheid:** Sommige toepassingen hebben een systeem nodig wat kan reageren op veranderende en onverwachte situaties bijvoorbeeld het stukgaan van een sensor.

**Efficiëntie:** Een intelligente oplossing kan de efficiëntie van een huidig systeem verbeteren met ten opzichte van een conventioneel systeem dezelfde middelen.

**Autonoom**: Een intelligente oplossing kan een dezelfde taak afhandelen als een conventioneel systeem, maar dan met minder supervisie en interactie van een mens.

**Onderhoudskosten:** Een intelligent systeem zou goedkoper kunnen zijn in onderhoud omdat het langer kan draaien zonder interactie van een mens.

**Geen alternatief:** Soms kan je niet om een intelligent systeem heen. Bijvoorbeeld het analyseren van niet lineaire correlaties, dit is niet aan te pakken met traditionele methodes.

**Voorspelbaarheid:** Bij een kritische toepassing is een voorspelbaar systeem gewenst, traditionele systemen zijn voorspelbaar omdat het programma geen grote dynamische wijzigingen doorvoert. Als een probleem wordt aangepakt met een neuraal netwerk heeft de programmeur de implementatie niet volledig in de hand. Dit resulteert in een onvoorspelbaar systeem.

## 3.3. Hedendaagse KI Embedded systemen

Bij KI wordt vaak gedacht aan futuristische apparaten, toch wordt er al heel lang gebruik gemaakt van kunstmatige intelligentie. Dit is zelfs zo geïntegreerd in het dagelijks leven dat KI gebruikt wordt zonder erbij stil te staan. Een voorbeeld is de spellingchecker die iedereen die documenten schrijft dagelijks gebruikt. De KI van vandaag is allemaal zwakke KI, sterke KI is op dit moment alleen nog een gedachtegang van experts die voorspellen hoever KI kan gaan.

In Computerspellen zit soms ook ver gevorderde KI de zogeheten KI Engine. Deze KI Engine maakt de beslissingen van de tegenstander in deze spellen. Bij strategische spellen als Civilization moeten beslissing worden gemaakt op basis van harde getallen(aantal goudstukken, aantal soldaten enz.). In schietspellen moet de KI beslissingen maken op basis van zaken die moeilijker in getallen zijn uit te drukken. Een donker hoekje biedt wel meer bescherming, maar soms is deze plek ook juist goed te beschieten. De KI Engine kan tijdens een gevecht er voor kiezen om afstand te nemen om overzicht op de situatie te krijgen, maar hij kan ook er voor kiezen om de speler te overrompelen door dicht bij de speler te gaan staan. KI in computerspellen is niet volledig gericht op winnen, de KI is meer gericht op het nabootsen van een menselijke tegenspeler. De mate waarin een KI gericht is op winnen is vaak instelbaar door de moeilijkheidsgraad van het spel in te stellen.[1]

Voorgaande voorbeelden zijn applicaties die zich bevinden op geavanceerde systemen die eigenlijk niet meer binnen de categorie Embedded systemen vallen. Een realistisch voorbeeld van intelligentie in een Embedded systeem is de Magnum Intelligent-Control.[9] Dit is een thermostaat die zichzelf aanpast aan de omgevingsvariabelen. Een nieuw geplaatste thermostaat kijkt de eerste paar dagen naar de opwarm karakteristieken van een huis. Aan de hand van deze statistieken kan dit systeem efficiënt het huis opwarmen naar de juiste temperatuur op het juiste tijdstip.

# 4. Kunstmatige intelligentie praktijk voorbeeld

Omdat autolearning onder KI valt is er in dit essay een praktisch voorbeeld opgenomen van een systeem wat zichzelf dingen kan aanleren, in dit geval winnen. NIM is een bekend spel wat moeilijk is te winnen voor een mens. Dit komt doordat er meerdere stappen vooruit gedacht moet worden. In het geval van de implementatie opgenomen in dit essay gaat het om een stapel van 21 stenen. Steeds mag een speler 1, 2 of 3 stenen wegpakken. Diegene die de laatste steen wegpakt heeft verloren. De tegenspeler wordt gerealiseerd op een ATMEGA128 die zelf leert. Aan het begin zal het eenvoudig zijn om te winnen van de ATMEGA maar uiteindelijk zal de ATMEGA bijna onverslaanbaar worden. Door de code zo compact mogelijk te schrijven wordt aangetoond dat het ook mogelijk is om kunstmatige intelligentie toe te passen op zeer kleine Embedded systemen.

Figuur 1: NIM abstract

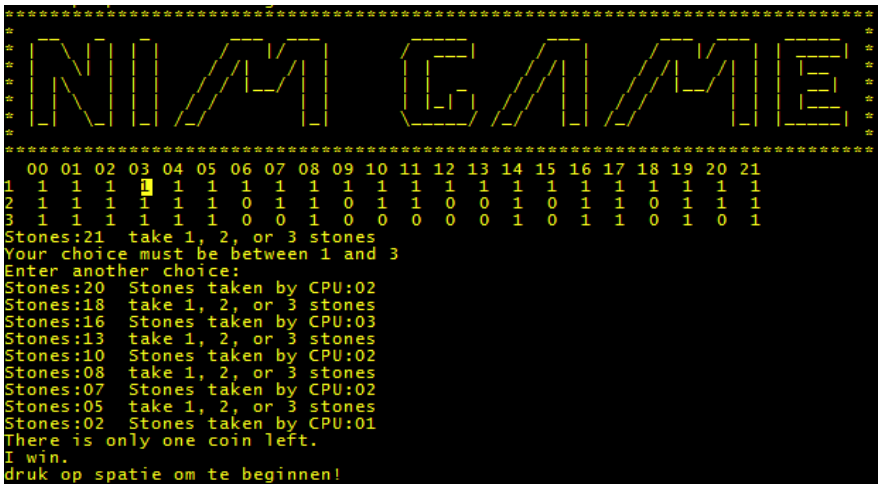
## 4.1. Opstelling

Het praktijkvoorbeeld is gemaakt op een ATMEGA128 ontwikkelingsbord, het grote voordeel hiervan is dat de hardware al compleet is. De software is ontwikkeld in AVRstudio en geschreven in C. De grafische kant van het spel wordt verzorgd door een terminal waar via UART, ASCII heen wordt gestuurd. Door het gebruik van de UART blijft er meer tijd over voor de kern van het programma. De ATMEGA128 wordt geprogrammeerd met een MyAVR SmartUSB2 van MyAVR.de.

## 4.2. Programmaloop

In de bijlagen van dit document is de volledige source code opgenomen, de loop van het programma is als volgt:

1. Initialiseren van de periferie en registers
2. Stuur het splash screen via de UART naar het scherm
3. Leg 21 stenen neer
4. Maak het “moves” register leeg
5. Stuur via de UART het aantal stenen naar het scherm
6. Vraag aan de gebruiker hoeveel stenen hij van de stapel wil pakken
7. Stuur via de UART het nieuwe aantal stenen naar het scherm
8. Laat de computer aan de hand van het aantal stenen en het leerregister kiezen hoeveel stenen hij van de stapel pakt
9. Zet deze beweging in het “moves” register
10. Spring terug naar stap 5 tot dat iemand de laatste steen pakt
11. Als de computer heeft verloren: zet in het leerregister welke foute bewegingen er zijn gemaakt, deze zijn te vinden in het “moves” register




Figuur 2: NIM Game implementatie

## 4.3. Intelligentie

Het spel vertoont intelligentie omdat hij leert van zijn fouten. In een tweedimensionale array is een plekje opgenomen voor elke mogelijke zet in het spel (leerregister). Elke zet die de computer doet wordt bijgehouden in een “moves” register. Als de computer het spel verliest heeft hij foute beslissingen genomen in het spel. De foute bewegingen worden als fout gemarkeerd in het leerregister. Bij het volgende spel zal de computer niet meer voor deze zet kiezen.

## 4.4. Bevindingen

De voornaamste bevinding van het praktijkonderzoek is dat C geen geschikte taal is voor KI. Waar C heel erg is gericht op stappen maken om bij een oplossing van een probleem te komen zijn talen als Prolog en SQL gericht op voorwaarden stellen waar een oplossing aan moet voldoen.[10] Een dergelijke logische programmeertaal zou in het voorgaande praktijkvoorbeeld toegepast kunnen worden bij het kiezen van 1, 2 of 3 stenen. Een andere niet verwachte bevinding is dat de code voor de intelligentie klein qua omvang is maar toch al ingewikkeld. Het intelligente deel van het programma is niet gericht op het winnen van het spel maar op het verliezen afleren. Dit bewijst dat een programmeur nauwelijks verstand hoeft te hebben van de toepassing (spel) maar slechts de code moet kunnen genereren die aanzet tot het autolearnen.

# 5. Conclusie

Tot op heden is er nog geen sterke KI bereikt. Toch gaan de ontwikkelingen steeds verder. Een goed voorbeeld hiervan is de chip van HP die in de buurt komt van een model van onze hersenen.

Het praktijkvoorbeeld opgenomen in het essay toont aan dat een vorm van zwakke KI eigenlijk op elk huidig Embedded systeem kan worden geïmplementeerd. De vraag blijft echter of deze simpele vorm van KI enige toegevoegde waarde heeft.
Tijdens het implementeren van het praktijkvoorbeeld bleek dat gestructureerde talen zoals C niet geschikt zullen zijn voor het beschrijven van echte intelligentie. Daarom zijn er logische talen zoals Prolog ontwikkeld. In de toekomst zullen deze talen mogelijk standaard worden opgenomen in opleidingen als Technische Informatica.

De laatste jaren is er in steeds meer systemen een bepaalde vorm van intelligentie toegevoegd met als doel het leven van de mens eenvoudiger te maken. Deze ontwikkeling zal zich de komende jaren ook voorzetten. Wij denken dat er ooit sterke KI wordt bereikt die ook in staat is om bijvoorbeeld rationeel te denken. Naast de ontwikkeling van sterke KI blijft nog altijd de vraag over of dit ethisch verantwoord is.

# 6. Toekomstvisie kunstmatige intelligentie en Embedded systemen

Het antwoord op de hoofdvraag van dit essay, “wat is de toekomst van kunstmatige intelligentie in Embedded systemen?” is in ieder geval dat de toekomst er voor KI rooskleurig uit ziet. Er zijn onderzoeksteams over de hele wereld bezig met het ontwikkelen van steeds geschiktere platformen en programmeertalen om sterke KI te kunnen realiseren. Mocht sterke KI ooit bereikt worden is het nog niet zeker dat het ook wordt ingezet. Zoals al eerder in dit essay wordt aangeven voorspellen sommige onderzoekers dat het mensenras ooit als huisdier worden gehouden door robots met kunstmatige intelligentie.

Onze visie op Kunstmatige Intelligentie wordt ondersteund door het merendeel van de mensen die iets te maken hebben met de ontwikkeling ervan. Zolang KI nog zorgt voor extra luxe in het dagelijks leven zal de ontwikkeling voortzetten. Het zal dan ook veelvuldig gaan opduiken in andere vakgebieden zoals transport en beveiliging. De tegenhangers van KI zijn vaak bang dat er uiteindelijk iets gecreëerd wordt dat ons als mensenras uitroeit. Maar zover zijn we echter nog lang niet.

# 7. Literatuurlijst


1. Artificial intellingence. Beschikbaar via; http://en.wikipedia.org/wiki/Artificial_intelligence Geraadpleegd op: 1 december 2010

2. David Wechsler. Intelligentie definitie. Beschikbaar via; http://en.wikipedia.org/wiki/David_Wechsler Geraadpleegd op: 1 december 2010

3. De Turningtest. Beschikbaar via: http://nl.wikipedia.org/wiki/Turingtest Geraadpleegd op: 1 december 2010

4. Jack Copeland. What is Artificial Intelligence. Beschikbaar via; http://www.alanturing.net/turing_archive/pages/Reference%20Articles/what_is_AI/What%20is%20AI02.html Geraadpleegd op: 1 december 2010

5. Bert van Dam. Kunstmatige Intelligentie. Breng u microcontroller tot leven! Geraadpleegd op: 15 december 2010

6. R. Stanley William. How we found the missing memristor. Beschikbaar via; http://spectrum.ieee.org/semiconductors/processors/how-we-found-the-missing-memristor/0 Geraadpleegd op: 15 december 2010

7. Massimiliano Versace en Ben Chandler. A mide made from memristors. Beschikbaar via; http://spectrum.ieee.org/robotics/artificial-intelligence/moneta-a-mind-made-from-memristors/0 Geraadpleegd op: 15 december 2010

8. Jacky Baltes ad John Anderson. Complex AI on Small Embedded Systems Humanoid Robotics using Mobile Phones. Beschikbaar via; http://aalab.cs.umanitoba.ca/~andersj/Publications/pdf/embeddedAI.pdf Geraadpleegd op: 16 december 2010

9. Magnum Heating. Beschikbaar via; http://www.magnumheating.nl/nl/producten/thermostaten/magnum-intelligent-control/ Geraadpleegd op: 16 december 2010

10. Prolog. Beschikbaar via; http://nl.wikipedia.org/wiki/Prolog Geraadpleegd via: 17 december 2010

# Bijlage 1 Source code NIM

```
//////////////////////////////////////////////////////
//                                                  //
//    nim.c                                         //
//      ATMEGA128 4.000.000 MHz     //
//                                                  //
//////////////////////////////////////////////////////

#include <stdlib.h>
#include <avr/io.h>
#include <avr/pgmspace.h>
#include <util/delay.h>
#include <math.h>
#include <avr/interrupt.h>

#define USART_BAUDRATE 19200
#define BAUD_PRESCALE (((F_CPU / (USART_BAUDRATE * 16UL))) - 1)

typedef enum { Human, Computer } player;
char decision_mem [22] [3]= {
{1, 1, 1}, {1, 1, 1}, {1, 1, 1},
{1, 1, 1}, {1, 1, 1}, {1, 1, 1},
{1, 1, 1}, {1, 1, 1}, {1, 1, 1},
{1, 1, 1}, {1, 1, 1}, {1, 1, 1},
{1, 1, 1}, {1, 1, 1}, {1, 1, 1},
{1, 1, 1}, {1, 1, 1}, {1, 1, 1},
{1, 1, 1}, {1, 1, 1}, {1, 1, 1},
{1, 1, 1}};  //19 steen situaties(21 t/m 2) en 3 zet mogenlijkheden, 0=niet doen, 1=mogelijke zet

char move_history[22]; //gezette moves in potje 0 als er niks is gedaan bij dat aantal stenen, 1,2 of 3 bij een zet.
//char moves=0;

void init(){
      DDRA   |=  (1<<PA0);      //PA0 als uitgang

      UCSR0B |= (1 << RXEN0) | (1 << TXEN0);   // Turn on the transmission and reception circuitry
      UCSR0C |= (1 << UCSZ00) | (1 << UCSZ01); // Use 8-bit character sizes

      UBRR0L = BAUD_PRESCALE; // Load lower 8-bits of the baud rate value into the low byte of the UBRR register
      UBRR0H = (BAUD_PRESCALE >> 8); // Load upper 8-bits of the baud rate value into the high byte of the UBRR register
}

void usart_putchar(char c){
        UDR0 = c;
        while(bit_is_clear(UCSR0A, TXC0));
        UCSR0A = _BV(TXC0);
}

char usart_getchar(){
        while ((UCSR0A & (1<<RXC0)) == 0){}
        return UDR0;
}
```

```
void usart_putstring(char s[]){
	int i;
	for(i=0;s[i] != '\0';i++){
	usart_putchar(s[i]);
	}
}

void usart_putstone(int s){
	usart_putchar((s/10)+0x30);
	usart_putchar((s%10)+0x30);
}
void zoslimbenik(){ //leer array uitprinten
	int i;

	usart_putchar(' ');
	for (i=0;i<=21;i++){
		usart_putchar(' ');
		usart_putstone(i);
	}
	usart_putstring("\r\n");

	usart_putchar('1');
	for (i=0;i<=21;i++){
		usart_putstring("  ");
		usart_putchar((decision_mem[i][0])+0x30);
	}
	usart_putstring("\r\n");

	usart_putchar('2');
	for (i=0;i<=21;i++){
		usart_putstring("  ");
		usart_putchar((decision_mem[i][1])+0x30);
	}
	usart_putstring("\r\n");

	usart_putchar('3');
	for (i=0;i<=21;i++){
		usart_putstring("  ");
		usart_putchar((decision_mem[i][2])+0x30);
	}
	usart_putstring("\r\n");
}

void zoslimwordtik(){ //move history uitprinten
	int i;

	for (i=0;i<=21;i++){
		usart_putstone(i);
		usart_putchar(' ');
		usart_putchar(move_history[i]+0x30);
		usart_putstring("\r\n");
	}
}
```

```
void splash(){
	usart_putstring("druk op spatie om te beginnen!\r\n");
	while (UDR0 != ' '){
	}

usart_putstring("*********************************************************************************
******\r\n");
	usart_putstring("*  __   _  _     ___  ___      ______     ___     ___  ___   ______  *\r\n");
	usart_putstring("* |  \\ | | |   /  |/  |    /  ___|    /  |   /  |/  | | |_____| *\r\n");
	usart_putstring("* |   \\| | |  / /|  /| |    | |       / /| |  / /|  /| | | |__   *\r\n");
	usart_putstring("* | |\\   | | | / /|__/ | |    | | _    / / | | / /|__/ | | |  __|  *\r\n");
	usart_putstring("* | | \\  | | |/ /     | |    | |_| | / /  | |/ /      | | | |___  *\r\n");
	usart_putstring("* |_|  \\_| |_|/_/      |_|    \\_____//_/   |_|/_/      |_| |_____| *\r\n");
	usart_putstring("*                                                        *\r\n");

usart_putstring("*********************************************************************************
******\r\n");
	zoslimbenik();
}

void AnnounceWinner(player whoseTurn)
{
	int i;

	usart_putstring("There is only one coin left.\r\n");
	switch (whoseTurn) {
	   case Human:    usart_putstring("I win.\r\n");
			  break;
	   case Computer: usart_putstring("I lose. but now i am smarter\r\n");
			  for(i=0;i<=21;i++){//0 zetten in decision_mem array
				switch(move_history[i]){
				case 3: //3 stenen gepakt, en dit was fout
					decision_mem[i][2]=0;
					break;
				case 2://2 stenen gepakt, en dit was fout
					decision_mem[i][1]=0;
					break;
				case 1://1 stenen gepakt, en dit was fout
					decision_mem[i][0]=0;
					break;
				case 0: //niks gedaan
					break;
				}
			}
			  break;
	}
}

int user_choose(int stones)//Interacts with the user, allowing a choice for the game of Nim
{
	int choice;
	usart_putstring("take 1, 2, or 3 stones\r\n");

	choice = usart_getchar()-0x30; //char ontvangen, int van maken
	while (choice <1 || choice > 3 || stones - choice <= 0)//Loop until a proper choice is made
	{
		if (choice < 1 || choice > 3) //Can only choose between 1 and 3
		{
			usart_putstring("Your choice must be between 1 and 3\r\n");
			usart_putstring("Enter another choice: \r\n");
```

```
                choice = usart_getchar()-0x30; //char ontvangen, int van maken
            }
            else if (stones - choice <= 0) //User has taken too many markers
            {
                usart_putstring("You must leave at least one stone\r\n");
                usart_putstring("Enter another choice: \r\n");
                choice = usart_getchar()-0x30; //char ontvangen, int van maken
            }
        }
        return choice;
}

int computer_choose(int stones)//Interacts with the user, allowing a choice for the game of Nim
{
        if(decision_mem[stones][2]==1 && stones >=4){ //mag ik er 3 pakken
                move_history[stones] = 3;
                return 3;
                }
        else if(decision_mem[stones][1]==1 && stones >=3){ //mag ik er 2 pakken
                move_history[stones] = 2;
                return 2;
                }
        else if(decision_mem[stones][0]==1&& stones >=2){ //mag ik er 1 pakken
                move_history[stones] = 1;
                return 1;
                }
        else{
                        decision_mem[stones][0] = 1;
                        decision_mem[stones][1] = 1;
                        decision_mem[stones][2] = 1;
                        return 1;
        }
}

int main(){

        int stones;
        int taken=0;
        player whoseTurn; //zie typedef
        int i;

        init();

        while(1){
                splash();
                stones = 21; //aantal begin stenen
                for(i=0;i<=22;i++){  //mo
                        move_history[i] = 0;
                }
                whoseTurn = Human; //eventueel kiesbaar maken
                while (stones > 1) {
                        usart_putstring("Stones:");
                        usart_putstone(stones);
                        usart_putstring("  ");

                        switch (whoseTurn) {
                            case Human:
                                taken = user_choose(stones);
                                whoseTurn = Computer;
                                break;
```

```
                    case Computer:
                        taken = computer_choose(stones);
                        usart_putstring("Stones taken by CPU:");
                        usart_putstone(taken);
                        usart_putstring("\r\n");
                        whoseTurn = Human;
                        break;
                }
                stones -= taken;
            }
            AnnounceWinner(whoseTurn);
        }
}
```