'Master mind' op de KIM

J. M. van der Peijl en D. M. de Boer

U kent waarschijnlijk wel het bekende Mastermind spel. Normaal wordt dit spel gespeeld door 2 personen, waarbij eigenlijk maar 1 persoon actief bezig is met het raden van een verborgen code. De tweede persoon heeft de wat passieve taak om met zwarte en witte pennetjes aan te geven of de gekozen kleuren goed zijn en of zij op de goede plaats staan. Vervang de kleuren door cijfers, de zwarte en witte pennetjes door letters en de KIM neemt de plaats in van de passieve speler. Hij kiest op uw commando een geheime kleurencombinatie en geeft met letters aan hoeveel cijfers juist gekozen zijn. Een speciale 'spiek knop' geeft de mogelijkheid om stiekum achter het schotje te kijken, de consequentie is wel dat de beurtenteller met 30 punten wordt opgehoogd.

Gebruiksaanwijzing

Als u het hele programma heeft ingetypt (adressen 0200 . . . 033D) kunt u op adres 0200 het programma starten.
Het display wordt dan donker en de KIM zal op uw commando steeds 1 cijfer (kleur) in zijn geheugen zetten. Na een aantal drukken op de knop is de geheime combinatie klaar en het display licht op. Bij het kiezen van de combinatie hebben we de volgende mogelijkheden.

enige keren drukken op toets nummer:	elk cijfer ligt tussen:	van de 4 cijfers zijn er dan max.:
0	0 . . . 6	4 hetzelfde
1	0 . . . 6	3 hetzelfde
2	0 . . . 6	2 hetzelfde
3	0 . . . 6	4 verschillende
4	1 . . . 6	4 verschillende
5	1 . . . 5	4 verschillende
andere toets	geen actie	

Op deze manier kan het spel dus in verschillende moeilijkheidsgraden worden gespeeld. Met het oplichten van het display geeft de KIM aan dat de combinatie in het geheugen staat.
De twee meest rechtse display's vormen de beurtenteller, op de 4 linkse displays verschijnt de door u geprobeerde combinatie. Mocht u een verkeerde toets hebben ingedrukt, dan kunt u de combinatie gewoon opnieuw intypen. Bij een druk op de E (enter) wordt de door u geprobeerde combinatie vergeleken met de geheime combinatie in het geheugen. De KIM antwoordt met een C voor elke goede kleur die niet op de goede plaats staat en met een A voor elke goede kleur die wel op de goede plaats staat. De plaats van de A's en de C's zeggen niets over plaats van de goede kleur. Als u de volgende combinatie wilt proberen, moet u eerst met een willekeurige toets de beurtenteller ophogen en het display schoonmaken. In hopeloze gevallen geeft een druk op knop 'C' de oplossing terwijl de beurtenteller met 30 opgehoogd wordt. Een goede oplossing geeft 'AAAA' met daarachter het aantal beurten.
Het display gaat een aantal malen aan en uit en blijft tenslotte donker. Nu kan een nieuwe geheime code gemaakt worden voor het volgende spelletje.

Beoordelingssystemen

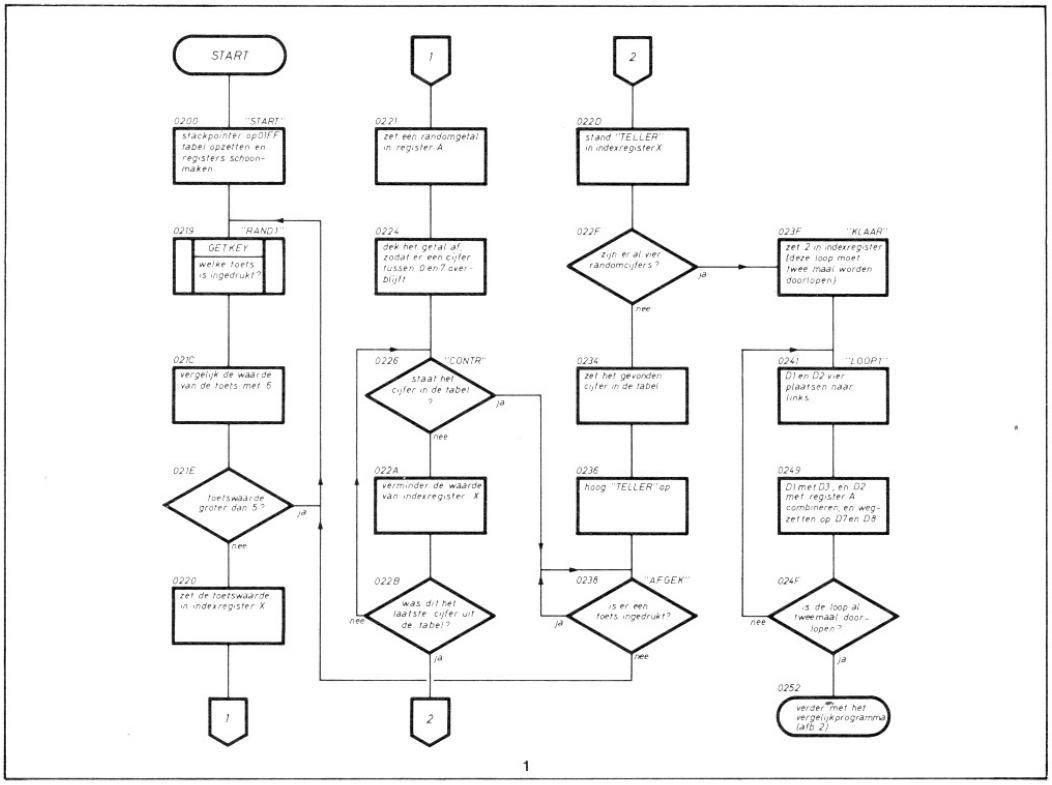
Nadat de redactie een aantal spelletjes op KIM gespeeld had bleek dat voor het zetten van de witte en zwarte pennetjes verschillende methoden bestaan.

Stel de situatie:
0113 geheime code
3331 geprobeerde code

KIM zal op deze situatie reageren met 2 x de letter C (2 witte pennetjes). Er zit in de geprobeerde code immers de 3 (op de verkeerde plaats) en ook de 1 is een goed cijfer. Er bestaat echter ook een systeem dat zegt dat de 3 drie maal goed is geraden, terwijl bovendien de 1 éénmaal goed is. Totaal moeten er dus 4 witte pennetjes komen. Bent u dit systeem gewend? De KIM past zich aan uw wensen aan, indien u de **code EA op adres 0336 zet.** (Normaal staat op dit adres code E4.) Een derde systeem zegt: Het cijfer 1 komt 2 x in de geheime code voor, de 3 komt 1 x in de geheime code voor, bij elkaar dus 3 witte pennetjes. **Code EA op adres 032E** en de KIM denkt net als u. (Normaal staat hier E6.).

De werking

Voor degene die geïnteresseerd is in de werking en opbouw van het programma volgt hier een beschrijving van dit Mastermind programma. Het programma valt uiteen in 2 belangrijke gedeelten, nl. de randomgenerator (afb. 1) die zorgt dat de KIM een geheime code kan kiezen zonder voorkeur voor bepaalde combinaties, en het vergelijkprogramma (afb. 2). Dit laatste programmadeel verzorgt de vergelijking van de geprobeerde code met de geheime code. We zullen deze twee delen apart behandelen.



1

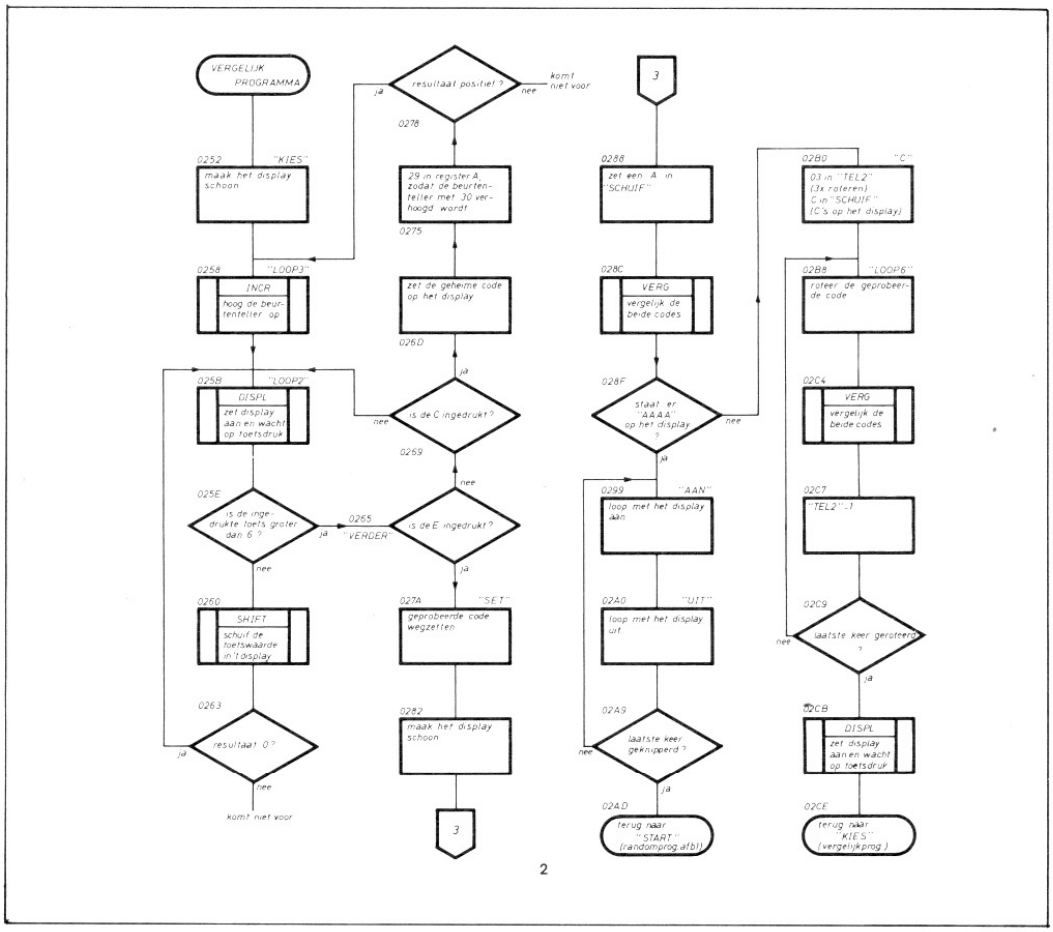
1 **De flowchart van de randomgenerator.**

De randomgenerator (principe)
Het principe van de randomgenerator is heel eenvoudig. Op het moment dat een toets wordt ingedrukt kijken we op welke stand de timer van de KIM staat. (Als u adres 1706 invoert dan ziet u de timer lopen.) Doordat het moment van drukken onafhankelijk is van de stand van deze teller, zal de verkregen waarde volkomen willekeurig zijn. Bij de randomgenerator voor dit spelletje worden er aan de verkregen getallen wel wat eisen gesteld. Het verkregen cijfer moet immers, afhankelijk van de ingedrukte toets, tussen de 0 en 6 of 1 en 5 liggen. Omdat al deze getallen niet meer dan 3 bits nodig hebben, kunnen we in ieder geval de eerste 5 bits van het verkregen random getal '0' maken. Hierdoor blijft een cijfer tussen de 0 en 7 (00000000 . . . 00000111) over. In het geheugen wordt nu een tabel van verboden cijfers geschreven en het verkregen cijfer wordt met de in de tabel voorkomende cijfers vergeleken. Wanneer een gelijkheid wordt gevonden, wordt het verkregen cijfer afgekeurd, en zal er een keer extra op een toets moeten worden gedrukt. De tabel ziet er als volgt uit:

Adres	Inhoud
00D5	06
00D4	00
00D3	FF
00D2	FF
00D1	FF
00D0	07

Indien we bij het kiezen van een randomcijfer op de '0' drukken zal het verkregen cijfer alleen worden vergeleken met de inhoud van adres 00D0, zodat alleen de 7 uitgesloten is. Het verkregen cijfer zal dus tussen 0 en 6 liggen. Als we bij het kiezen van een randomcijfer op de '3' drukken, zal het verkregen cijfer in een loop vergeleken worden met de inhoud van de adressen 00D3 . . . 00D0.

Is het eerste cijfer gekozen, stel dat dit een 3 is, dan wordt dit cijfer vergeleken met FF, FF, FF en 07. Omdat geen gelijkheid wordt gevonden, wordt het getal goedgekeurd en op adres 00D1 onthouden. Het getal wordt dus in de tabel van verboden cijfers geschreven, en de 3 kan geen tweede keer meer voorkomen. Stel dat bij de tweede druk op de '3' een 1 gevonden wordt, deze 1 wordt nu vergeleken met FF, FF, 03 en 07. Ook de 1 wordt goedgekeurd en op adres 00D2 onthouden. Als we bij de derde druk op '3' weer een 1 krijgen, wordt hij afgekeurd. De 1 wordt nu immers vergeleken met FF, 01, 03 en 07, zodat een gelijkheid wordt geconstateerd. Indien een goed derde cijfer is gevonden wordt dit op adres 00D3 gezet. Het 4de cijfer wordt nog wel vergeleken met de tabel, maar omdat er geen 5de cijfer meer komt is het niet nodig dit 4de cijfer in de tabel te zetten. Hadden we inplaats van op de '3' op de '5' gedrukt, dan werden de verkregen getallen ook nog vergeleken met '0' en '6', zodat ook deze cijfers uitge-



2

sloten zouden zijn. Het zal duidelijk zijn dat we voor het krijgen van een geheime code minstens 4 x op de 0, 1, 2, 3, 4 of 5 moeten drukken. Voor elk getal dat afgekeurd wordt moeten we nog een keer extra drukken. De schakeling laat niet zien wanneer een getal afgekeurd is. Wel laat zij zien wanneer de geheime code compleet is. De microprocessor verlaat dan het randomprogramma en begint aan het vergelijkprogramma. Het display licht op, met de beurtenteller op 1, ten teken dat u uw eerste code kunt invoeren.

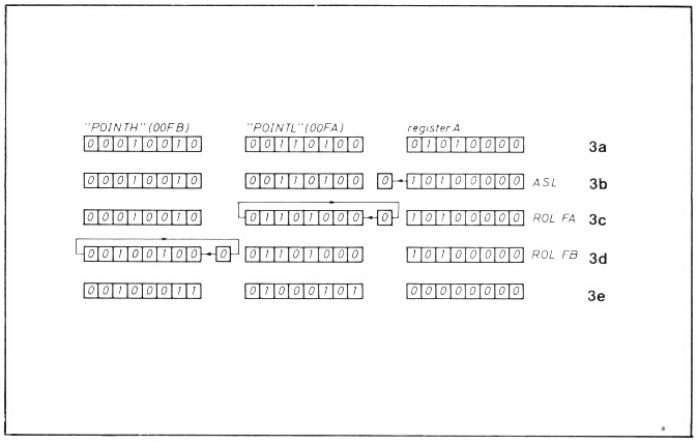
De flowchart van het randomprogramma

De flowchart van het randomprogramma zien we in afb. 1. Op adres 200 beginnen we met het goed zetten van verschillende tellers. Ook de tabel op de plaatsen 00D0 t/m 00D5 wordt door dit deel opgezet. De volgende stap is het bepalen welke toets is ingedrukt (adres 0219). Is er geen toets ingedrukt dan krijgt register A de waarde $ 15 ($ betekent weer: hexa-decimaal). We zien dat de microprocessor steeds dezelfde loop doorloopt totdat een van de getallen 0 . . . 5 worden ingedrukt. De waarde van de toets wordt in indexregister X gezet (adres 0220). Vervolgens wordt de stand van de timer in register A gezet, en met 00000111 afgedekt, zodat een getal tussen 0 en 7 overblijft. Op adres 0226 begint de controleloop. Afhankelijk van de inhoud van indexregister X (dus de waarde van de ingedrukte toets) wordt het verkregen cijfer vergeleken met een groot of klein deel van de tabel. Als een gelijkheid wordt gevonden, springt de microprocessor naar 'AFGEK' adres 0238. Hier wordt gewacht tot de toets

2 De flowchart van het vergelijkprogramma.

is losgelaten. Zodra dit het geval is wordt weer naar het begin van het programma gesprongen. Indien in de controleloop (0226) geen gelijkheid wordt gevonden, kijkt de microprocessor naar de teller (adres 00D6) om te zien of we al 4 randomcijfers hebben. Is dit niet het geval, dan wordt het gevonden cijfer op de juiste plaats in de tabel gezet, de teller wordt opgehoogd en zodra de toets wordt losgelaten beginnen we weer bij 'rand 1' om het volgende randomcijfer te vinden. Zodra we 4 cijfers hebben gaat de microprocessor op adres 0241 de cijfers combineren. Op dit moment staan de eerste drie cijfers op de adressen 00D1 . . . 00D3, en het 4de cijfer staat in regis-

ter A. Om te laten zien hoe het combineren in zijn werk gaat zullen we als voorbeeld aannemen dat op de plaatsen 00D1 . . . 00D3 de getallen 01 . . . 03 staan en in register A 04. Op adres 023F (programma, lijst 1) wordt in indexregister X de waarde 02 gezet. Op de adressen 0241 . . . 0247 wordt de inhoud van adres 00D0 + X = 00D2 4 plaatsen naar links geschoven. Het getal 02 is nu dus 20 geworden. Op adres 0249 wordt de inhoud van diezelfde geheugenplaats opgeteld bij de inhoud van register A. We krijgen dus 4 + 20 = 24. Het resultaat 24 wordt nu op adres 00D6 + X = 00D8 gezet. In register A komt nu de inhoud van adres 00D3 te staan (03). Indexregister X wordt met 1 verminderd (dit wordt 01) en de loop wordt voor de tweede maal doorlopen. Nu wordt de inhoud van adres 00D1 geschoven, en bij de inhoud van register A opgesteld. Het resultaat wordt 13. Deze waarde komt op adres 00D7 te staan. Indexregister X wordt weer verminderd, met als resultaat 00. Hierdoor zal niet gesprongen worden naar 'loop 1', en de microprocessor begint aan het vergelijkprogramma.



3a De toestand van de registers nadat register A 4 x naar links geschoven is. De inhoud van het displaygeheugen is '1234'.
3b De instructie ASL, bit 7 schuift in het Carry-bit.
3c De instructie ROL FA, bit 7 van register A welke in het Carrybit stond schuift in bit 0 van 'POINTL', terwijl bit 7 van 'POINTL' weer in het Carrybit schuift.
3d De instructie ROL FB, bit 7 van 'POINTL' komt nu in bit 0 van 'POINTH'.
3e De toestand na 4 x roteren, de inhoud van het displaygeheugen is nu '2345'.

Vergelijkprogramma (principe)
Het vergelijkprogramma valt ook weer uiteen in verschillende delen. Allereerst is er het invoerdeel. Met dit stukje programma is het mogelijk uw code in te voeren. Alleen getallen tussen 0 en 6 kunnen worden ingevoerd. Steeds als een nieuw cijfer wordt ingetoetst, schuiven de reeds aanwezige cijfers op het display een plaats naar links (net als bij het invoeren van een adres).

Om dit te bereiken moet het gehele getal dat bestaat uit 2 x 8 bits 4 bits naar links worden geschoven. (1 hexa-decimaal cijfer bestaat uit 4 bits, 1 cijfer naar links komt dus overeen met 4 bits naar links.) Dit gebeurt in de subroutine 'SHIFT'. Het is alleen mogelijk het invoerdeel te verlaten door een druk op de 'C' of de 'E'. Met een druk op 'C' wordt de geheime code op POINTLO en POINTHI gezet, waardoor ze later in het programma zichtbaar worden op het display. Tevens wordt in register A het getal 29 gezet, en naar de ophoogsubroutine gesprongen. Deze subroutine set de 'decimal mode flag' waardoor er decimaal wordt gerekend. Ook de carry wordt geset, zodat het uiteindelijke resultaat is dat de beurtenteller met 30 wordt opgehoogd. Wanneer we van de subroutine terug keren, beginnen we weer met het invoerdeel, zodat weer nieuwe codes kunnen worden ingevoerd. Als we op de 'E' drukken, verlaten we het invoerdeel, en beginnen met het programmadeel dat de A's en de C's op het display zet. Dit programmadeel is gebaseerd op de subroutine 'VERG'. Deze subroutine doet niets anders dan het in verticale zin vergelijken van de getallen. Bij een geconstateerde gelijkheid wordt een A of C in het display geschoven met behulp van de eerder besproken subroutine 'SHIFT'. We zullen dit aan de hand van een voorbeeld duidelijk maken. Stel, we hebben:

4341 geheime code
4313 geprobeerde code.

We zetten nu $ 0A op adres 00E2 en springen naar de subroutine 'VERG'. Deze subroutine constateert 2 gelijkheden, nl. in de 1e cijfers en in de 2e cijfers (de 4 en de 3). Hierdoor zal ook 2 x de inhoud van adres 00E2 in het display geschoven worden (er staan dus nu twee A's). Om te voorkomen dat de gelijkheden nog een keer herkend worden, moeten de getallen worden veranderd, zodanig dat zij gegarandeerd anders zijn dan alle voorkomende cijfers. Bij de cijfers uit de geheime code tellen we 8 op, zodat een cijfer tussen de 8 en E (= 14) ontstaat.

De cijfers uit de geprobeerde code worden vervangen door 'F' (= 15). Als we terug komen van de subroutine zien de codes er als volgt uit:

CB41 geheime code
FF13 geprobeerde code.

In de volgende stap gaan we kijken of er 'schuine' gelijkheden zijn (zoals de '1'). Allereerst zetten we nu 0C op adres 00E2, en we laten de geprobeerde code 1 plaats roteren. De codes zien er nu zo uit:

CB41 geheime code
F13F geprobeerde code.

We springen weer naar dezelfde subroutine, en er wordt geen gelijkheid geconstateerd. Het proces wordt nog een keer herhaald. Na rotatie krijgen we:

CB41 geheime code
13FF geprobeerde code.

Ook hier wordt geen gelijkheid geconstateerd. Hadden we bij de eerste vergelijking echter niets afgedekt, zou er hebben gestaan:

4341 geheime code
1343 geprobeerde code

met het gevolg dat 2 gelijkheden geconstateerd worden en dat er ten onrechte 2 C's op het display zouden verschijnen. Hieruit blijkt dus de noodzaak van het afdekken. De laatste keer roteren geeft:

CB41 geheime code
3FF1 geprobeerde code.

Nu wordt het 4e cijfer als gelijk herkend (de '1'), en de inhoud van adres 00E2 (nu 'C') wordt in het display geschoven. Nog een keer roteren heeft geen zin meer, omdat we dan weer in de uitgangspositie komen. Het eindresultaat op het display is dus '0AAC'. Tot slot springt de microprocessor weer naar de display subroutine. In wezen wordt hier pas het display aangezet, en wordt '0AAC', dat al wel in het display geheugen stond, zichtbaar gemaakt. Pas na een druk op een van de toetsen verlaat de microprocessor deze subroutine en beginnen we weer vooraan het vergelijkprogramma met het schoonmaken van het display en het ophogen van de beurtenteller. Alleen wanneer er 4 A's op het display verschijnen verlaten we het vergelijkprogramma, en beginnen we vooraan het randomprogramma zodat een nieuw spelletje kan worden gespeeld.

De flowchart van het vergelijkprogramma (afb. 2)

Het eerste deel van de flowchart spreekt voor zichzelf. Op adres 025B springt de microprocessor naar subroutine 'DISPL', en hij blijft daar totdat een toets is ingedrukt. Alle indrukken tussen 0 en 6 hebben tot gevolg dat de microprocessor via adres 025E en 0260 aankomt bij de subroutine 'SHIFT', waar het naar links schuiven van de cijfers op het display verzorgd wordt, terwijl op het meest rechtse display het nieuwe cijfer wordt gezet. Als de subroutine weer verlaten wordt is **altijd** het Z-bit uit het status register '1'. Hiervan maken we gebruik op adres 263, waar we via een BEQ (spring als Z = 1) nu onvoorwaardelijk terug kunnen springen naar 'loop 2' (025B). Bij alle ingedrukte toetsen, groter dan 6 en ongelijk aan E of C gaat de microprocessor via 0265 en 0269 weer terug naar 'loop 2', en wordt er verder geen actie ondernomen. Wordt een 'E' ingedrukt, dan verlaat de microprocessor het invoerdeel en begint op adres 027A aan het vergelijk van de codes. Om te beginnen moet de geraden code, die nu in het displaygeheugen staat, in andere geheugenplaatsen worden gezet. Het displaygeheugen (POINTLO en POINTHI) moet straks immers worden gebruikt om de A's en de C's in op te bergen. Op adres 0282 wordt het displaygeheugen met nullen gevuld, zodat later op de plaatsen waar geen A of C staat een '0' komt te staan. Nu begint het vergelijken. Eerst zetten we een A op 'schuif' (adres 00E2) omdat we beginnen met het vergelijken van getallen die op de goede plaats staan. Op adres 028C springt de microprocessor naar de subroutine 'VERG', en wanneer hij hiervan terugkeert staat er voor alle cijfers die op de goede plaats stonden een A in het displaygeheugen. Bovendien zijn de gelijkheden vervangen door andere cijfers, zoals besproken bij het principe van het vergelijkprogramma.

Als er geen 'AAAA' in het displaygeheugen staat, springt de microprocessor naar 'C' (adres 02B0), waar we beginnen met 03 op adres 00E1 te zetten.

Deze geheugenplaats dient als teller, en houdt bij hoeveel keer de geraden code is geroteerd. (We moeten 3 x roteren.) Dit roteren begint op adres 02B8. Op adres 02C4 worden de codes dan weer vergeleken. Vervolgens wordt de teller met 1 verminderd. Dit hele proces wordt 3 x herhaald, en bij elke geconstateerde gelijkheid wordt er een 'C' in het display geschoven. Het principe van dit roteren is gelijk aan het principe van de subroutine 'SHIFT', welke nog besproken zal worden. Wanneer de vergelijkloop is doorlopen, springen we weer naar subr. 'DISPL'.

Hier worden de A's en de C's eigenlijk pas zichtbaar gemaakt. De microprocessor blijft in deze subroutine totdat een toets wordt ingedrukt. Op dat moment wordt het vergelijkgedeelte opnieuw gestart (adres 0252). Het display wordt op '0000' gezet, de beurtenteller opgehoogd, en de geheime code wordt opnieuw op de adressen 00E5 en 00E6 gezet, omdat deze code in de vorige vergelijkloop is verminkt. (De cijfers werden immers veranderd om te voorkomen dat dubbele gelijkheden worden gevonden, zie 'principe vergelijk-

1 Het programma van de randomgenerator.

Lijst 1

0200	D8	START	CLD-*impl*		D nul maken.
0201	EA		NOP-*impl*		
0202	A2 FF		LDX-*imm*	SFF	Stackpointer
0204	9A		TXS-*impl*		op 01FF.
0205	86 D1		STX-*Z page*	TABEL+1	
0207	86 D2		STX-*Z page*	TABEL+2	
0209	86 D3		STX-*Z page*	TABEL+3	
020B	E8		INX-*impl*		
020C	86 D4		STX-*Z page*	TABEL+4	Tabel opzetten.
020E	A0 07		LDY-*imm*	$07	
0210	84 D0		STY-*Z page*	TABEL	
0212	88		DEY-*impl*		
0213	84 D5		STY-*Z page*	TABEL+5	
0215	86 D6		STX-*Z page*	TELLER	'TELLER' en de
0217	86 F9		STX-*Z page*	INH	beurtenteller schoonmaken.
0219	20 6A 1F	RAND 1	JSR-*abs*	GETKEY	Welke toets?
021C	C9 06		CMP-*imm*	$06	Indien groter dan 5
021E	10 F9		BPL-*rel*	RAND1	naar 'RAND1'.
0220	AA		TAX-*impl*		Toets naar X.
0221	AD 06 17		LDA-*abs*	TIMER	Randomcijfer in A.
0224	29 07		AND-*imm*	$07	Cijfer afdekken.
0226	D5 D0	CONTR	CMP-*Zp, X*		Vergelijk het
0228	F0 OE		BEQ-*rel*	AFGEK	gevonden cijfer
022A	CA		DEX-*impl*		met de tabel.
022B	10 F9		BPL-*rel*	CONTR	Laatste vergelijking?
022D	A6 D6		LDX-*Z page*	TELLER	Stand teller in X.
022F	EO 03		CPX-*imm*	$03	Indien er 4
0231	FO OC		BEQ-*rel*	KLAAR	getallen zijn naar 'KLAAR'.
0233	EA		NOP-*impl*		Zet het gevonden
0234	95 D1		STA-*Zp, X*	TABEL+1	cijfer in de tabel.
0236	E6 D6		INC-*Z page*	TELLER	Hoog de teller op.
0238	20 FE 1E	AFGEK	JSR-*abs*	AK	Wacht tot de toets
023B	DO FB		BNE-*rel*	AFGEK	is losgelaten.
023D	FO DA		BEQ-*rel*	RAND1	Naar 'RAND1'.
023F	A2 02	KLAAR	LDX-*imm*	$02	Begin combineerloop.
0241	16 DO	LOOP 1	ASL-*Zp, X*	TABEL	
0243	16 DO		ASL-*Zp, X*	TABEL	Cijfers 4 plaatsen
0245	16 DO		ASL-*Zp, X*	TABEL	naar links.
0247	16 DO		ASL-*Zp, X*	TABEL	
0249	75 DO		ADC-*Zp, X*	TABEL	Cijfers combineren.
024B	95 D6		STA-*Zp, X*	TELLER	Combinatie wegzetten.
024D	A5 D3		LDA-*Z page*	TABEL+3	
024F	CA		DEX-*impl*		Eind combineerloop.
0250	DO EF		BNE-*rel*	LOOP1	

programma'.) Op adres 025B ('loop 2') beginnen we dan weer met het invoerdeel, en de nieuwe code kan worden geprobeerd. Als de code goed is geraden, zullen op adres 028F vier A's op het display worden herkend. Hierdoor zal de microprocessor op adres 0299 beginnen aan een loop waarin het display brandt en op adres 02A0 een loop waarin het display uit is. Dit geheel wordt 6 x herhaald, zodat het display 6 x aan en uit gaat. Het principe van de loops zal wel duidelijk zijn, geheugen 00D4 wordt hier gebruikt als teller. Deze teller begint op 00 en telt naar beneden via FF weer naar 00. Omdat deze teller steeds met '00' uit een loop komt, hoeft er aan het begin van een loop niet steeds '00' ingeschreven te worden. Teller 00D5 houdt het aantal knipperingen bij. Wanneer het display voor de laatste keer uit is gegaan, springen we weer naar het begin van het hele programma, en kan er weer een nieuwe geheime code gemaakt worden.

2 Het vergelijkprogramma.
3 De gebruikte subroutines.

Lijst 2

0252	A9 00	KIES	LDA-*imm*	$00	Display schoonmaken.
0254	85 FA		STA-*Z page*	POINTL	
0256	85 FB		STA-*Z page*	POINTH	
0258	20 D1 02	LOOP3	JSR-*abs*	INCR	Beurtenteller ophogen.
025B	20 E2 02	LOOP2	JSR-*abs*	DISPL	Welke toets?
025E	10 05		BPL-*rel*	VERDER	Indien groter dan 6 naar 'VERDER'.
0260	20 F7 02		JSR-*abs*	SHIFT	Schuif het display door en spring terug.
0263	F0 F6		BEQ-*rel*	LOOP2	
0265	C9 0E	VERDER	CMP-*imm*	$0E	Indien 'E' is ingedrukt naar 'SET' springen.
0267	F0 11		BEQ-*rel*	SET	
0269	C9 0C		CMP-*imm*	$0C	Indien 'C' niet is ingedrukt, naar 'LOOP2'.
026B	D0 EE		BNE-*rel*	LOOP2	
026D	A5 D7		LDA-*Z page*	GEHLO	Zet de geheime code op het display.
026F	85 FA		STA-*Z page*	POINTL	
0271	A5 D8		LDA-*Z page*	GEHHI	
0273	85 FB		STA-*Z page*	POINTH	
0275	A9 29		LDA-*imm*	$29	Zet 29 in reg.A, en ga via 'LOOP3' naar de subroutine 'INCR'
0277	EA		NOP-*impl*		
0278	10 DE		BPL-*rel*	LOOP3	
027A	A5 FA	SET	LDA-*Z page*	POINTL	Het geprobeerde getal wegzetten.
027C	85 E7		STA-*Z page*	GEPRLO	
027E	A5 FB		LDA-*Z page*	POINTH	
0280	85 E8		STA-*Z page*	GEPRHI	
0282	A9 00		LDA-*imm*	$00	Maak het display schoon.
0284	85 FA		STA-*Z page*	POINTL	
0286	85 FB		STA-*Z page*	POINTH	
0288	A9 0A		LDA-*imm*	$0A	'A' in 'SCHUIF', zodat er A's op het display komen, en spring naar 'VERG'.
028A	85 E2		STA-*Z page*	SCHUIF	
028C	20 06 03		JSR-*abs*	VERG	
028F	A9 AA		LDA-*imm*	$AA	Staan er 4 A's op het display? Zo nee, spring naar 'C'.
0291	C5 FA		CMP-*Z page*	POINTL	
0293	D0 1B		BNE-*rel*	C	
0295	C5 FB		CMP-*Z page*	POINTH	
0297	D0 17		BNE-*rel*	C	
0299	20 1F 1F	AAN	JSR-*abs*	SCANDS	Display aan.
029C	C6 D4		DEC-*Z page*	TABEL+4	
029E	D0 F9		BNE-*rel*	AAN	
02A0	A0 00	UIT-	LDY-*imm*	$00	Display uit.
02A2	88	LOOP4	DEY-*impl*		
02A3	D0 FD		BNE-*rel*	LOOP4	
02A5	C6 D4		DEC-*Z page*	TABEL+4	
02A7	D0 F7		BNE-*rel*	UIT	
02A9	C6 D5		DEC-*Z page*	TABEL+5	Laatste keer geknipperd?
02AB	D0 EC		BNE-*rel*	AAN	Zo nee, naar 'AAN'.
02AD	4C 00 02		JMP-*abs*	START	Terug naar 'START'.
02B0	A9 03	C	LDA-*imm*	$03	'TEL2' houdt bij hoeveel maal geroteerd wordt.
02B2	85 E1		STA-*Z page*	TEL2	
02B4	A9 0C		LDA-*imm*	$0C	'C' in 'SCHUIF', zodat er C's op het display komen.
02B6	85 E2		STA-*Z page*	SCHUIF	
02B8	A0 04	LOOP6	LDY-*imm*	$04	Vier maal schuiven.
02BA	A5 E8	LOOP5	LDA-*Z page*	GEPRHI	Alle cijfers in de geprobeerde code 1 plaats naar links.
02BC	0A		ASL-*accu*		
02BD	26 E7		ROL-*Z page*	GEPRLO	
02BF	26 E8		ROL-*Z page*	GEPRHI	
02C1	88		DEY-*impl*		
02C2	D0 F6		BNE-*rel*	LOOP5	
02C4	20 06 03		JSR-*abs*	VERG	Schuif zonodig een C op het display.
02C7	C6 E1		DEC-*Z page*	TEL2	
02C9	D0 ED		BNE-*rel*	LOOP6	Laatste maal geroteerd?
02CB	20 E2 02		JSR-*abs*	DISPL	Wacht op toetsindruk en spring terug.
02CE	4C 52 02		JMP-*abs*	KIES	

Lijst 3

02D1	F8	INCR	SED-*impl*		Tel decimaal de inhoud van reg. A op bij 'INH'.
02D2	38		SEC-*impl*		
02D3	65 F9		ADC-*Z page*	INH	
02D5	85 F9		STA-*Z page*	INH	
02D7	D8		CLD-*impl*		
02D8	EA		NOP-*impl*		
02D9	A5 D7		LDA-*Z page*	GEHLO	Copieer de geheime code.
02DB	85 E5		STA-*Z page*	COPLO	
02DD	A5 D8		LDA-*Z page*	GEHHI	
02DF	85 E6		STA-*Z page*	COPHI	
02E1	60		RTS-*impl*		Terug.
02E2	20 1F 1F	DISPL	JSR-*abs*	SCANDS	Wacht tot de toets wordt losgelaten.
02E5	D0 FB		BNE-*rel*	DISPL	
02E7	20 1F 1F	LOOP8	JSR-*abs*	SCANDS	Wacht tot de toets wordt ingedrukt.
02EA	F0 FB		BEQ-*rel*	LOOP8	
02EC	20 1F 1F		JSR-*abs*	SCANDS	Is de toets wel ingedrukt?
02EF	F0 F6		BEQ-*rel*	LOOP8	
02F1	20 6A 1F		JSR-*abs*	GETKEY	Welke toets is ingedrukt?
02F4	C9 07		CMP-*imm*	$07	Kleiner dan 7?
02F6	60		RTS-*impl*		Terug.
02F7	0A	SHIFT	ASL-*accu*		Cijfer 4 plaatsen naar links schuiven.
02F8	0A		ASL-*accu*		
02F9	0A		ASL-*accu*		
02FA	0A		ASL-*accu*		
02FB	A0 04		LDY-*imm*	$04	Vier maal schuiven.
02FD	0A	LOOP7	ASL-*accu*		Afb. 3b
02FE	26 FA		ROL-*Z page*	POINTL	Afb. 3c
0300	26 FB		ROL-*Z page*	POINTH	Afb. 3d
0302	88		DEY-*impl*		Indien nog geen 4 maal, naar 'LOOP7'.
0303	D0 F8		BNE-*rel*	LOOP7	
0305	60		RTS-*impl*		Terug.
0306	A9 F0	VERG	LDA-*imm*	$F0	Vergelijk het linker cijfer.
0308	85 E0		STA-*Z page*	MODE	
030A	20 15 03		JSR-*abs*	VERG2	
030D	A9 0F		LDA-*imm*	$0F	Vergelijk het rechter cijfer.
030F	85 E0		STA-*Z page*	MODE	
0311	20 15 03		JSR-*abs*	VERG2	
0314	60		RTS-*impl*		Terug.
0315	A2 02	VERG2	LDX-*imm*	$02	Wordt 2 maal doorlopen.
0317	B5 E4	LOOP9	LDA-*Zp, X*	COPIE	Haal twee cijfers v.d.geheime code, en zet er één op E3 of E4.
0319	25 E0		AND-*Z page*	MODE	
031B	95 E2		STA-*Zp, X*	SCHUIF	
031D	B5 E6		LDA-*Zp, X*	COPHI	Haal het overeenkomstige cijfer van de geprobeerde code en vergelijk.
031F	25 E0		AND-*Z page*	MODE	
0321	D5 E2		CMP-*Zp, X*	SCHUIF	
0323	F0 04		BEQ-*rel*	AFDEK	Indien gelijk naar 'AFDEK'.
0325	CA	VERG3	DEX-*impl*		Indien nog geen twee maal doorlopen naar 'LOOP9'.
0326	D0 EF		BNE-*rel*	LOOP9	
0328	60		RTS-*impl*		Terug.
0329	B5 E6	AFDEK	LDA-*Zp, X*	COPHI	Dek een cijfer uit de geheime code af met 'F'.
032B	05 E0		ORA-*Z page*	MODE	
032D	95 E6		STA-*Zp, X*	COPHI	
032F	A9 88		LDA-*imm*	$88	Afh. van 'MODE' komt er 08 of 80 in reg. A.
0321	25 E0		AND-*Z page*	MODE	
0333	15 E4		ORA-*Zp, X*	COPIE	Dek een cijfer uit de geprobeerde code af met '8'.
0335	95 E4		STA-*Zp, X*	COPIE	
0337	A5 E2		LDA-*Z page*	SCHUIF	Schuif een 'A' of een 'C' in het display.
0339	20 F7 02		JSR-*abs*	SHIFT	
033C	F0 E7		BEQ-*rel*	VERG3	Terug naar 'VERG3'.

De subroutines

Het programma maakt gebruik van 4 subroutines, nl. 'INCR', deze hoogt de beurtenteller op, 'DISPL', deze activeert het display en bepaalt welke toets is ingedrukt, 'SHIFT' zorgt dat een cijfer welke in register A staat, in het display wordt geschoven, en tot slot 'VERG', welke de codes vergelijkt. We zullen de subroutines nog even afzonderlijk behandelen.

INCR (adres 02D1, lijst 3)

Omdat de instructie 'INC' niet decimaal werkt, moeten we voor het ophogen van de beurtenteller gebruik maken van de optelinstructie 'ADC'. Omdat de beurtenteller decimaal moet werken wordt eerst de 'decimal mode flag' geset. Ook de carry wordt geset. Het eindresultaat op de beurtenteller wordt dus: inhoud register A + oude inhoud beurtenteller + 1. Wanneer we de teller met 1 willen ophogen, moet in register A dus de waarde 00 staan. Deze subroutine verzorgt tevens het copiëren van de geheime code naar de plaatsen 00E5 en 00E6. Het zal de kritische lezer opgevallen zijn dat deze subroutine slechts éénmaal wordt aangeroepen. In feite was het dus helemaal niet nodig dit stukje programma als subroutine uit te voeren. Bij een uitbreiding van het programma, door een inventieve geest, kan echter handig gebruik gemaakt worden van deze subroutine door b.v. het toekennen van strafpunten.

DISPL (adres 02E2, lijst 3)

Deze subroutine zorgt er steeds voor dat de cijfers, welke in het displaygeheugen staan (POINTLO, POINTHI, INH), op het display zichtbaar worden. Tevens wordt bepaald of een toets is ingedrukt, en zo ja welke. Voor het zichtbaar maken van de gegevens maakt deze subroutine gebruik van een subroutine in het monitorprogramma (SCANDS). Als de microprocessor terug komt van 'SCANDS' heeft register A de waarde '00' zolang er geen toets is ingedrukt. De eerste loop loopt van adres 02E2 . . . 02E6. In deze loop wordt gewacht totdat er geen toets meer is ingedrukt. Dit is nodig omdat de vorige toets meestal nog ingedrukt is. Was er geen toets ingedrukt, of de vorige toets wordt losgelaten, dan komt de microprocessor in de 2e loop (02E7 . . . 02EB). Hier wordt gewacht totdat er een toets wordt ingedrukt. Zodra dit het geval is wordt op de adressen 02EC . . . 02F0 nog een keer gecontroleerd of er echt een toets ingedrukt was. Het is nl. ook mogelijk dat de 2e loop verlaten wordt door een stoorpiekje. Op adres 02F1 wordt bepaald welke toets is ingedrukt. Tot slot wordt de toets met $07 vergeleken, zodat het N-bit in het statusregister wordt geset voor de toetsen tussen 0 en 6. Later kunnen hier dan weer voorwaardelijke sprongen mee gemaakt worden.

SHIFT (adres 02F7, lijst 3)

De werking van deze subroutine zullen we aan de hand van een voorbeeldje uitleggen. Stel dat op het display staat: 1234, en in register A de waarde 05. Allereerst wordt op de adressen 02F7 . . . 02FA register A 4 maal naar links geschoven. Het resultaat wordt: 50. In afb. 3 zien we nu wat er achtereenvolgens gebeurt. Afb. 3 a laat de situatie zien zoals hij is nadat register A 4 x geschoven is. Op adres 02FD wordt register A nog een keer geschoven (afb. 3b). Het bit dat uit register A schuift komt in het 'carry' bit van het status register te staan. Bij de eerste ROL instructie (02FE en afb. 3c) wordt dit bit in geheugenplaats 'POINTLO' geschoven, terwijl het bit dat hier naar buiten wordt geschoven, weer in de carry komt te staan. Bij de ROL instructie op adres 300 (afb. 3d) wordt dan dit bit weer in 'POINTHI' geschoven. Op deze manier is het geheel 1 plaats naar links geschoven. Dit proces wordt in een loop vier maal herhaald, zodat aan het eind van deze subroutine alle bits 4 plaatsen zijn opgeschoven (afb. 3e).

VERG (adres 0306, lijst 3).

In 'VERG' worden 2 codes met elkaar vergeleken. De subroutine is te verdelen in 3 belangrijke gedeelten, nl. het frame, op de adressen 0306 . . . 0314, het vergelijkdeel op de adressen 0315 . . . 0328 en het afdekdeel op de adressen 0329 . . . 033D. In het vergelijkdeel wordt steeds één van de twee geheugenplaatsen van de geheime code vergeleken met het overeenkomstige geheugen van de geraden code. Omdat er op één geheugenplaats 2 cijfers staan, moet er steeds 1 cijfer worden afgedekt. We moeten immers maar 1 cijfer tegelijk vergelijken. In het framedeel wordt eerst op adres 00E0 de waarde $F0 gezet. Het vergelijkdeel gebruikt deze waarde later om via de 'AND' instructie het rechtse cijfer 0 te maken, zodat het linker cijfer vergeleken kan worden. Op adres 030A springen we voor de eerste keer naar het vergelijkdeel op adres 0315. Omdat dit deel 2 x doorlopen moet worden zetten we eerst in register X de waarde 02. Vervolgens wordt een deel van de geheime code in register A gezet (de inhoud van adres 00E6), en afgedekt met F0 (de inhoud van adres 00E0). Het gevolg is dat er van de totaal 4 cijfers er nu één in register A staat. Dit cijfer wordt zolang op adres 00E4 gezet. We zijn inmiddels beland op adres 031D van het programma. Hier wordt een overeenkomstig deel van de geraden code in register A gezet en afgedekt (00E8). Op adres 0221 worden de cijfers vergeleken. Hierna wordt het vergelijkdeel nog een keer doorlopen. Nu wordt echter het linker cijfer van de geheugens 00E5 en 00E7 vergeleken.

Als dit gebeurd is, springen we terug naar het framedeel (030D). Nu zetten we geen $F0 maar $0F op adres 00E0, en springen weer naar het vergelijkdeel. Het hele proces wordt herhaald, maar nu voor het rechter cijfer. Wanneer een gelijkheid wordt gevonden springen we naar het afdekdeel op adres 0329. Hier wordt eerst het betreffende cijfer van de geheime code afgedekt. Op adres 032F en 0331 wordt in register A de waarde $80 of $08 gezet. Met de 'ORA' instructie wordt deze 8 dan opgeteld bij het desbetreffende cijfer van de geraden code. Aan het eind van dit deel wordt nog de inhoud van adres 00E2 ('A' of 'C') met behulp van subroutine 'SHIFT' op het display gezet.

Uitbreidingen

Met dit programma kunt u een aardig spelletje spelen. Maar probeer ook eens kleine dingen aan het programma te veranderen. U kunt b.v. eens proberen i.p.v. A en C de E en de F te gebruiken als codepinnetjes. U kunt de beurtenteller zo maken dat hij niet verder telt dan 99, zodat er niet gesmokkeld kan worden met de 'C'-toets (3 x drukken op deze toets en de beurtenteller wijst 10 punten minder aan).

Voor de gevorderde knutselaar: Probeer eens voor elkaar te krijgen dat na elke 30 sec. bedenktijd de beurtenteller automatisch opgehoogd wordt, dat brengt wat spanning in het spel. U kunt ook nog iets maken dat automatisch de score van verschillende spelers bijhoudt. Het voordeel van de programmeer-hobby is dat een instructie u niets kost, en dat u geen transistortjes op kunt blazen met verkeerde instructies.