‘Mastermind on the KIM’

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

You are probably familiar with the well-known game Mastermind. Normally this game is played by two people, with only one person actively trying to guess a hidden code. The second person has the passive task of indicating, with black and white pegs, whether the chosen colours are correct and whether they are in the right positions. Replace the colours with digits, the black and white pegs with letters, and the KIM takes the place of the passive player. At your command, it chooses a secret colour combination and uses letters to indicate how many digits have been correctly guessed. A special “cheat button” makes it possible to peek secretly behind the screen; the consequence, however, is that the turn counter is increased by 30 points.

Instructions

Once you have entered the entire program (addresses 0200 ... 03D), you can start it at address 0200. The display will then go dark, and the KIM will store one digit (colour) in its memory each time you give the command. After a number of button presses, the secret combination is complete and the display lights up.

When choosing the combination, the following options are available:

Number of presses on the numbered key	Each digit is between	|Maximum among the 4 digits
0	0..6	4 identical
1	0..6	3 identical
2	0..6	2 identical
3	0..6	4 different
4	1..6	4 different
5	0..6	4 identical
Other key	No action	

In this way, the game can be played at different levels of difficulty. When the display lights up, the KIM indicates that the combination is stored in memory.

The two rightmost displays form the turn counter; the four leftmost displays show the combination you have guessed. If you have pressed the wrong key, you can simply enter the combination again. Pressing E (enter) compares your guessed combination with the secret combination in memory. The KIM responds with a C for each correct colour that is not in the right position, and an A for each correct colour that is in the right position. The positions of the A's and C's do not indicate the positions of the correct colours. If you want to try the next combination, first press any key to increase the turn counter and clear the display.

In hopeless cases, pressing key C reveals the solution, while increasing the turn counter by 30. A correct solution displays "AAA", followed by the number of turns.

The display switches on and off a number of times and finally remains dark. A new secret code can now be made for the next game.

Scoring systems

After the editors had played a number of games on the KIM, it turned out that there are different methods for placing the white and black pegs.

Consider this situation:
0113 — secret code
3331 — guessed code

In this situation, the KIM responds with the letter C twice (two white pegs). The guessed code contains a 3 (in the wrong position), and the 1 is also a correct digit. There is, however, another system which says that the 3 has been guessed correctly three times, but the 1 once. This gives one correct digit in total, and therefore four white pegs. [The wording in the scan/OCR is unclear here.]

Are you used to this system? The KIM can adapt to your preference: set **code EA at address 0336.** (Normally address 036 contains code E4.) A third system says that the digit 1 occurs twice in the secret code and the digit 3 occurs once, making three white pegs in total. Set **code EA at address 032E** and the KIM will think as you do. (Normally E6 is found here.)

How it works

For those interested in the operation and construction of the program, a description of the Mastermind program follows. The program is divided into two important parts: the random-number generator (fig. 1), which makes the KIM choose a secret code without favouring particular combinations, and the comparison program (fig. 2). The latter part compares the guessed code with the secret code. We will discuss these two parts separately.

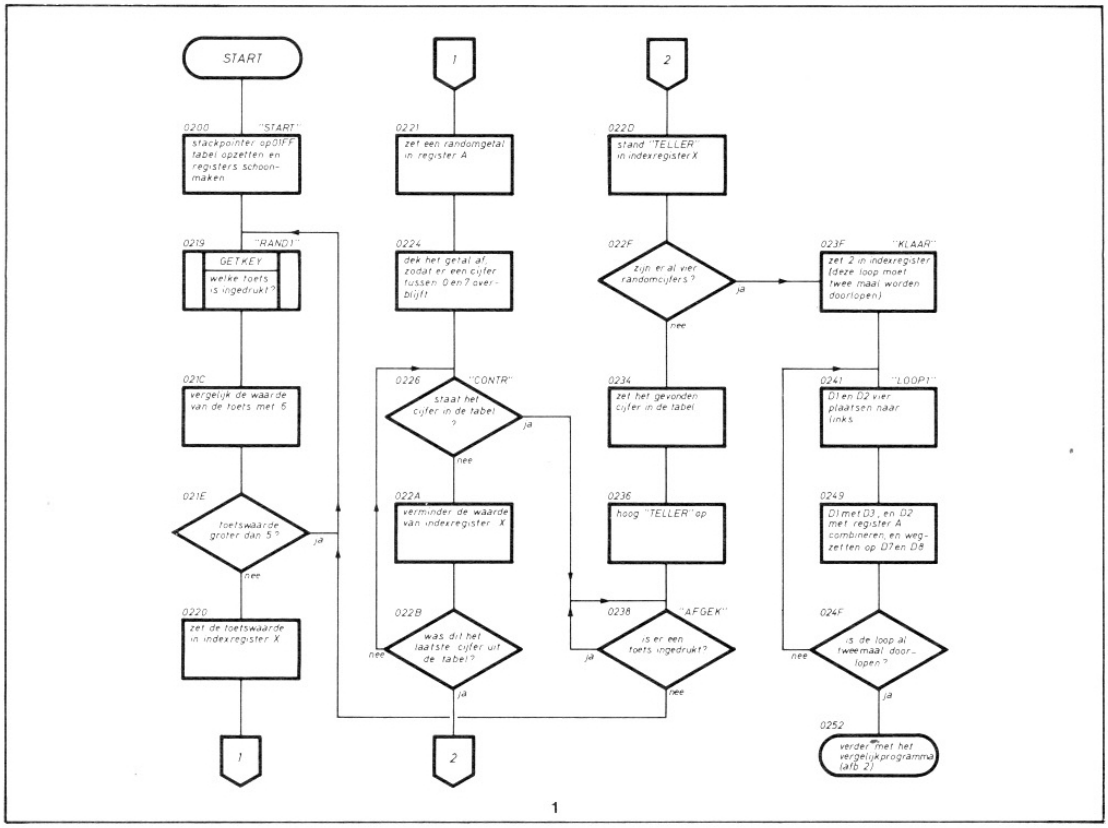


Figure 1, The random-number generator

RANDOM-NUMBER GENERATOR — PRINCIPLE

When a key is pressed, the program reads the KIM's timer (enter address 1706 to see it running). Since the press timing is independent of the timer, the value is treated as random. The resulting digit must, depending on the key, be between 0 and 6 or between 1 and 5. Since these values require only three bits, the first five bits of the random value are cleared, leaving 0–7 (00000000 ... 00000111). A table of forbidden digits is stored in memory, and each generated digit is checked against it. A match means rejection, requiring another key press:

Address	contents
00D5	06
00D4	00
00D3	FF
00D2	FF
00D1	FF
00D0	07

When choosing with key 0, only 7 (at 00D0) is excluded, leaving 0–6. With key 3, the candidate is compared in a loop with 00D3 … 00D0. Suppose the first digit is 3: it is compared with FF, FF, FF and 07, accepted, and stored at 00D1. It is added to the forbidden table, so 3 cannot recur. If the next press produces 1, it is compared with FF, 03 and 07, accepted, and stored at 00D2. If 1 is produced again, it is rejected because the table now contains 01, 03 and 07. The third accepted digit is stored at 00D3. The fourth is checked but need not be added to the table. Choosing with key 5 also excludes 0 and 6. At least four presses of keys 0–5 are needed; rejected values require extra presses. Rejections are not shown, but completion is. The processor then enters the comparison program; the display lights with the turn counter at 1.

RANDOM-PROGRAM FLOWCHART

At 0200 the counters and table 00D0–00D5 are initialized. At 0219 the pressed key is determined. If none is pressed, A is set to $15 ($ denotes hexadecimal), and the loop repeats until a key 0–5 is pressed. The key value is put in X at 0220. The timer value is placed in A and masked with 00000111, leaving 0–7. The checking loop starts at 0226. Depending on X, the candidate is compared with the appropriate part of the table. A match branches to AFGEK at 0238, waits for release, and returns to the program start.

If no match is found, the counter at 00D6 is checked. If fewer than four digits have been accepted, the digit is stored, the counter incremented, and after key release the program returns to “rand 1”. After four digits, the program combines them at 0241. The first three digits are at 00D1–00D3 and the fourth is in A. For example, if these locations contain 01, 02, 03 and A contains 04, X is set to 02 at 023F. The value at 00D2 is shifted four places left, turning 02 into 20. At 0249 it is added to A: 4 + 20 = 24, stored at 00D8. A is then loaded from 00D3 (03), X is decremented to 01, and the loop repeats. The contents of 00D1 are shifted and added, producing 13, stored at 00D7. X becomes 00, so the loop ends and comparison begins.

COMPARISON PROGRAM — PRINCIPLE

Only digits 0–6 can be entered. Each new digit shifts the digits already on the display one place left. The 16-bit display value (four hexadecimal digits) is shifted by four bits using SHIFT. The input section can be left only with C or E. Pressing C copies the secret code to POINTO and POINTH so it can be displayed, loads 29 into A, and calls the increment routine. This sets decimal mode and carry, increasing the turn counter by 30. Input then resumes. Pressing E leaves input and starts comparison, which displays A’s and C’s using VERG and SHIFT.

Example:

4341 secret code
4313 guessed code

Put 0A at 00E2 and call VERG. It finds two matches in the first and second digits (4 and 3), shifting two A's into the display. To prevent recounting, matching secret-code digits are increased by 8 (making values 8–E), and matching guessed-code digits are replaced by F:

CB41 secret code
FF13 guessed code

Next, diagonal matches are checked. Put 0C at 00E2 and rotate the guessed code one place:

CB41
F13F

No match. Rotate again:

CB41
13FF

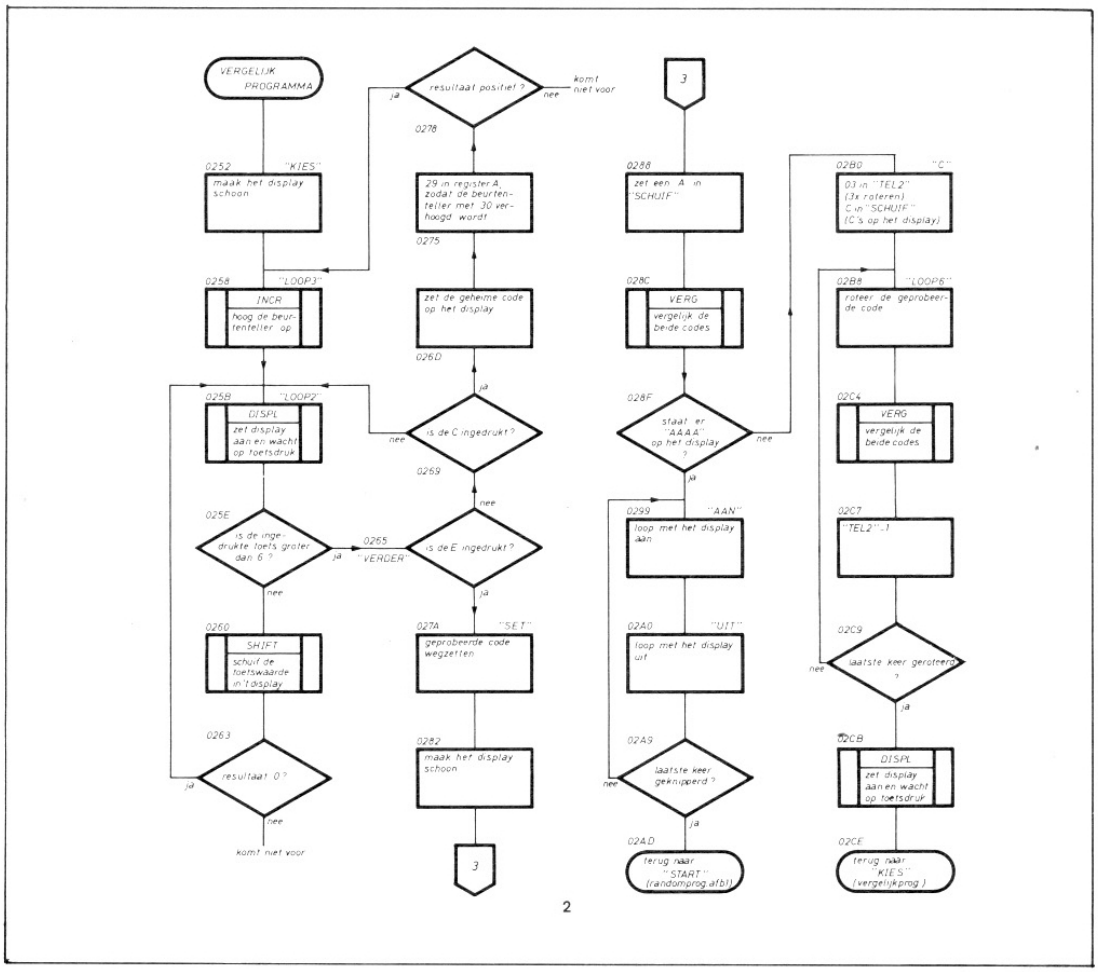
Again no match. Without masking during the first comparison, the codes would have been:

4341
1343

This would falsely produce two C's. The final rotation is:

CB41
3FF1

The fourth digit (1) matches, so C is shifted into the display. Another rotation would return to the starting position. The result is "0AAC". The display subroutine makes this stored result visible. After a key press, comparison restarts, the display is cleared and the turn counter incremented. Four A's mean the game is over and the main program restarts for a new secret code.

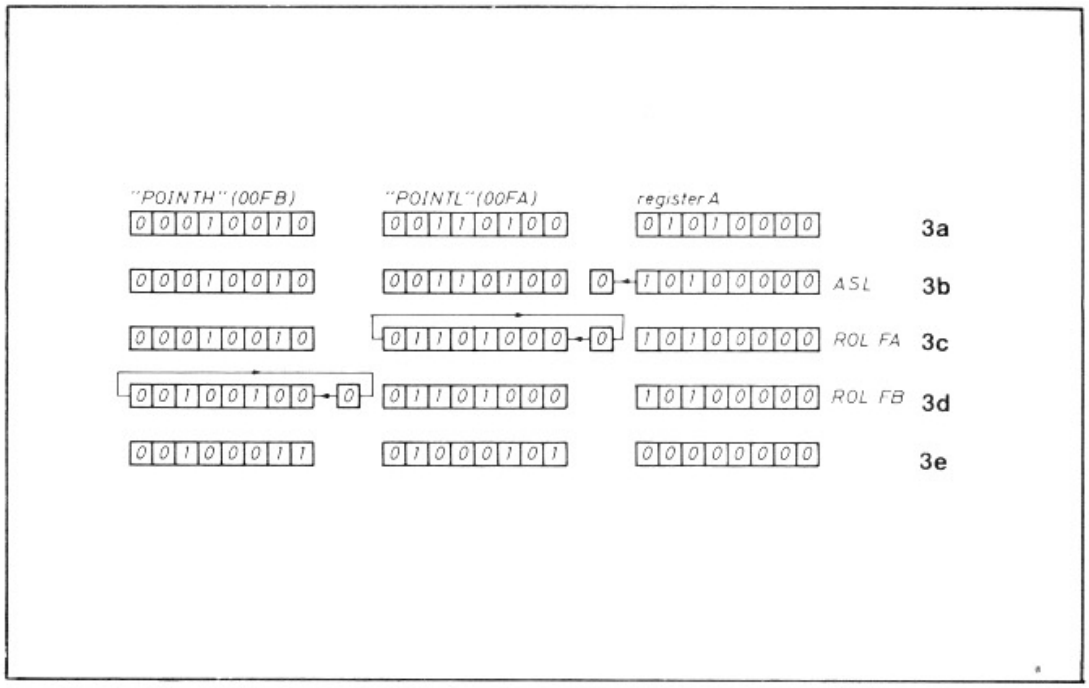


COMPARISON-PROGRAM FLOWCHART

At 025B the processor calls DISPL and waits for a key. Keys 0–6 go via 025E and 0260 to SHIFT, which shifts the display left and puts the new digit at the right. On return, the Z bit is set; BEQ at 0263 returns to loop 2 (025B). Keys greater than 6, except E and C, return via 0265 and 0269 without action. E leaves input and begins comparison at 027A. The guessed code is copied from display memory because POINTLO and POINTHI will hold the A's and C's. At 0282 display memory is cleared to zero. An A is placed in SHIFT at 00E2, and VERG is called at 028C to mark correctly positioned digits. Matches are altered to prevent duplicate detection.

If display memory does not contain "AAA", the program jumps to C at 02B0 and puts 03 at 00E1. This counter records the number of rotations. Rotation starts at 02B8; the codes are compared at 02C4 and the counter decremented. The process repeats three times, adding a C to the display for each match. Then the program returns to DISPL. A key press restarts the comparison section at 0252: display memory is set to 0000, the turn counter is incremented, and the original secret code is restored at 00E5 and 00E6 because the previous comparison altered it. Input resumes at 025B. If the code is

correct, four A's are detected at 028F. The display is lit in a loop at 0299 and turned off in a loop at 02A0. This is repeated six times. Location 00D4 is a counter that counts down from 00 through FF to 00; 00D5 counts the flashes. After the final flash, the program restarts and a new secret code can be created.



DISPLAY-SHIFT DIAGRAMS (FIG. 3)
3a — Register state after A has been shifted left four times; display memory contains 1234.
3b — ASL: bit 7 shifts into Carry.
3c — ROL FA: the Carry bit shifts into bit 0 of POINTL; bit 7 of POINTL returns to Carry.
3d — ROL FB: bit 7 of POINTL enters bit 0 of POINTH.
3e — After four rotations, display memory contains 2345.

THE SUBROUTINES
The four subroutines are INCR (increments the turn counter), DISPL (activates the display and identifies the key), SHIFT (shifts a digit in A into the display), and VERG (compares the codes).

INCR (address 02D1
INC does not operate in decimal mode, so ADC is used to increment the turn counter. Set the decimal-mode flag first, and set Carry as well. The value added is A + the old counter value + 1. To increment by one, A must contain 00. The routine also copies the secret code to 00E5 and 00E6. It is called only once, so a subroutine is not strictly necessary, though extensions such as penalty points could use it.

DISPL (address 02E2

This routine continually displays the digits in POINTLO and POINTHI and determines whether, and which, key is pressed. It uses the monitor routine SCANDS. On return from SCANDS, A is 00 if no key is pressed. The first loop (02E2–02E6) waits until no key is held, since the previous key may still be down. The second loop (02E7–02EB) waits for a press. Addresses 02EC–02F0 check again that a key was really pressed, since a disturbance could also exit the loop. Address 02F1 identifies the key. Comparing it with $07 sets the N bit for keys 0–6, which can then be used for conditional branches.

SHIFT (address 02F7

Suppose the display shows 1234 and A contains 05. At 02F7–02FA, A is shifted left four times, producing 50. At 02FD A is shifted once more; the bit shifted out enters Carry. The first ROL (02FE) shifts that bit into POINTLO, while POINTLO's outgoing bit enters Carry. ROL at 0300 shifts that bit into POINTHI. The value has moved one place left. Repeating this in a loop shifts all bits four places, producing display value 2345.

VERG (address 0306

VERG compares two codes. It has three sections: frame (0306–0314), comparison (0315–0328), and masking (0329–033D). Each memory location contains two digits, so one digit must be masked at a time. The frame first stores $F0 at 00E0. The comparison section uses AND with this value to zero the right-hand digit and compare the left-hand digit. At 030A it jumps to 0315; since this section runs twice, X is set to 02. A part of the secret code (from 00E6) is loaded into A and masked with F0, leaving one digit, temporarily stored at 00E0. At 031D the corresponding part of the guessed code (00E8) is loaded and masked; the digits are compared at 0321. The section runs again to compare the left digits from 00E5 and 00E7.The program returns to the frame at 030D, stores $0F rather than $F0 at 00E0, and repeats the process for the right-hand digit. A match branches to the masking section at 0329. The relevant secret-code digit is masked. At 032F and 0331, A is set to $80 or $08. ORA adds this bit to the corresponding guessed-code digit. Finally, the value at 00E2 (A or C) is shifted into the display with SHIFT.

EXTENSIONS

Try changing the program—for example, use E and F instead of A and C as the code pegs. You could make the turn counter stop at 99, preventing cheating with the C key (pressing it three times makes the counter show 10 fewer points).

For advanced users: make the turn counter increase automatically after every 30 seconds of thinking time, adding tension to the game. You could also make the program keep scores for different players automatically. Programming is a hobby in which instructions cost nothing, and incorrect instructions cannot blow up transistors.

Lijst 1

```
0200  D8        START   CLD-impl
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  E0 03              CPX-imm     $03      } Indien er 4
0231  F0 0C              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  D0 FB              BNE-rel     AFGEK    } is losgelaten.
023D  F0 DA              BEQ-rel     RAND1      Naar 'RAND1'.
023F  A2 02    KLAAR     LDX-imm     $02        Begin combineerloop.
0241  16 D0    LOOP 1    ASL-Zp, X   TABEL    }
0243  16 D0              ASL-Zp, X   TABEL    } Cijfers 4 plaatsen
0245  16 D0              ASL-Zp, X   TABEL    } naar links.
0247  16 D0              ASL-Zp, X   TABEL    }
0249  75 D0              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  D0 EF              BNE-rel     LOOP1
```

Lijst 2

```
0252  A9 00     KIES    LDA-imm     $00
0254  85 FA             STA-Z page  POINTL   } Display
0256  85 FB             STA-Z page  POINTH   } schoonmaken.

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

Lijst 3

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