GMC-4

Der **Gakken GMC-4** ist ein kompletter vom PC unabhängig zu nutzender kleiner Lerncomputer Er basiert auf dem Science Fair Microcomputer Trainer von Radio Shack aus den 80ern.

Dezember 2014 gab es das Franzis Lernpaket „Spielecomputer selbst programmieren“ mit dem GMC-4 und einem 96seitigen Handbuch von F. Kainka für nur 20€, so dass ich mir auch einen solchen kleinen Computer zugelegt habe.

Systembeschreibung

Bild, kurze Beschreibung

technische Daten

Merkmal	Beschreibung
CPU	EM61001
ROM	-
RAM	128 Byte a 4 Bit
Takt	4 MHz
Anzeige	7 Bit LED, 1 stellig Siebensegment-Anzeige
Tastatur	20 Tasten Hexadezimaltastatur
Peripherie	-
Software	-

Im Science Fair Microcomputer Trainer wird ein spezieller Microcontroller der TMS1000-Controller genutzt, speziell ein TMS1312. Dieser wurde mit 400 MHz betrieben.

Literatur

Downloads

- Anleitung, ..

Bedienung

0	
1	ラ
2	シ
3	ド

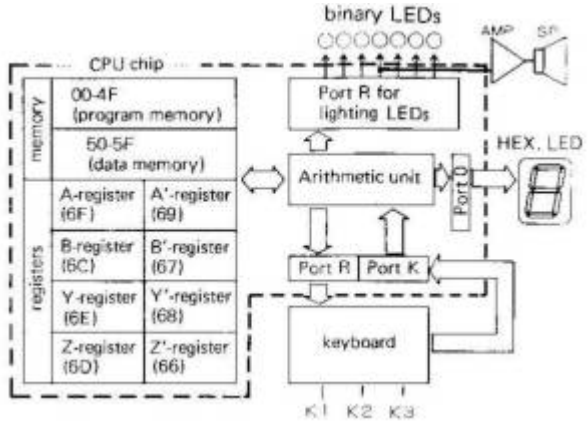
4	レ
5	ミ
6	ファ
7	ソ
8	ラ
9	シ
A	ド
B	レ
C	ミ
D	ファ
E	ソ
F	

Sonstiges

Prozesssor

Es wird ein spezieller 4-Bit-Prozessor simuliert.

- 4 Register A, B, X, Y
- 3 Ports P (LEDs), O (Siebensegmentanzeige), K (Tastatur)
- 80 Byte (a 4 Bit) Programm-Speicher (RAM-Bereich 00h..4Fh)
- 16 Byte Datenspeicher (RAM-Bereich 50h..5Fh)
- 1 Flag. Der Befehl JUMP ist ein bedingter Sprung und wird nur ausgeführt, wenn das Flag 1 ist
- 4 Wechselregister A', B', X', Y'



Der Befehlssatz

Der simulierte 4-Bit-Prozessor versteht 16 Maschinenbefehle.
Die Befehle sind 1..3 Byte (a 4 Bit) lang.

n = 0..F

Code	Befehl	Wirkung	Flag	Kommentar
0	KA (Key to A)	A := K	0, 1	Tastendruck nach A übernehmen, Flag = 0 bei Tastendruck, sonst Flag = 1
1	AO (A to Output)	O := A	1	A an Siebensegmentanzeige ausgeben
2	CH (Change)	A ⇔ B Y ⇔ Z	1	Tausche A und B sowie Y und Z
3	CY (Change A and Y)	A ⇔ Y	1	Austausch der Inhalte von A und Y
4	AM (A to Memory)	M := A	1	A in RAM-Adr (50+Y) kopieren
5	MA (Memory to A)	A := M	1	RAM-Adr (50+Y) in A zurücklesen
6	M+	A := M+A	0, 1	RAM-Adr (50+Y) zu A addieren, bei Überlauf: Flag = 1
7	M-	A := M-A	0, 1	RAM-Adr (50+Y) von A subtrahieren, bei Überlauf: Flag = 1
8 n	TIA n (Transfer Immediate to A)	A := n	1	Konstante in A laden
9 n	AIA n (Add Immediate to A)	A := A+n	0, 1	Konstante zu A addieren, bei Überlauf: Flag = 1
A n	TIY n (Transfer Immediate to Y)	Y := n	1	Konstante in Y laden
B n	AIY n (Add Immediate to Y)	Y := Y+n	0, 1	Konstante zu Y addieren, bei Überlauf: Flag = 1
C n	CIA n (Compare Immediate with A)	A == n ?	0, 1	A mit Konstante vergleichen, bei Übereinstimmung: Flag =0
D n	CIY n (Compare Immediate with Y)	Y == n ?	0, 1	Y mit Konstante vergleichen, bei Übereinstimmung: Flag =0
E n	CAL n (Call)	—	—	Unterprogrammaufrufe, Erweiterte Befehle
F n n	JUMP n n	Adr := nn	1	Direkter Sprung zur Adresse nn (High, Low) wenn Flag = 1

Unterprogramme:

Code	Befehl	Flag	Kommentar
E 0	CAL RSTO (Reset Port O)	1	Siebensegmentanzeige löschen
E 1	CAL SETR (Set Port R)	1	Einzelne LED einschalten. In Y wird die Nummer der LED (0-6) übergeben.
E 2	CAL RSTR (Reset Port R)	1	Einzelne LED ausschalten. In Y wird die Nummer der LED (0-6) übergeben.
E 4	CAL CMPL (Complement)	1	Komplement des A-Registers (aus F wird 0)
E 5	CAL CHNG	1	Inhalte der Register A,B,Y,Z mit A',B',Y',Z' tauschen
E 6	CAL SIFT	0, 1	A-Register bitweise nach recht schieben. Flag wird 1 wenn das rechte Bit 0 war.
E 7	CAL ENDS	1	Ende-Sound
E 8	CAL ERRS	1	Error-Sound
E 9	CAL SHTS (Short Sound)	1	Kurzer Ton
E A	CAL LONS	1	Langer Ton
E B	CAL SUND	1	Note spielen, die im A-Register übergeben wird (1 … E)
E C	CAL TIMR	1	(A + 1) * 0,1 Sekunden warten
E D	CAL DSPR (Display on Port R)	1	Ausgabe der RAM-Adressen 5F (high) und 5E (low) an die LEDs