

MMbg – Backgammon für MMBasic

Handbuch · Version 1.10.36 · Stand Oktober 2026

© 2026 Matthias Hollatz · Freie Software unter der GNU GPL v3 oder neuer

Ein Paket, drei Fassungen, jeder RP2350. Alle drei Programme haben die Version **1.10.36** und spielen dasselbe Spiel mit denselben Engines:

`mmbg_pc3.bas` – Boards **mit PSRAM**: **Pico Computer 3 (PC3)**, auch der **Colour Maximite 2** (Kapitel 15).
`mmbg_pc1.bas` – HDMI- und VGA-Boards **ohne PSRAM**, z. B. das PicoMite HDMI/USB-Board („PC1“) oder ein Pico 2 mit VGA (Kapitel 16).
`mmbg_pica.bas` – der **PicoCalc** (Teil 2).

Getestet mit PicoMite **V7.00.00RC6** und CMM2 **7.00.00b4**. Speicherangaben (mit ~ markiert) können sich mit späterer Firmware etwas ändern.

Inhalt

Teil 1 – MMbg PC3, Colour Maximite 2, PC1 und die gemeinsamen Grundlagen

1. Was ist MMbg?
2. Voraussetzungen
3. Installation
4. Der Bildschirm und das Terminal
5. Ein Match spielen
6. Undo, Speichern und Fortsetzen
7. Aufzeichnung, Spielanalyse und Bildschirmfotos
8. Engines und Spielstärke
9. Die neuronalen Netze
10. Spielen gegen gnubg über WLAN
11. Lange Testläufe (Long Runs)
12. Optionen
13. Befehlsübersicht
14. Dateien
15. MMbg auf dem Colour Maximite 2
16. MMbg PC1: HDMI und VGA ohne PSRAM
17. Hinweise und Grenzen
18. Ein Flug durch den Quellcode

Teil 2 – MMbg PiCa (PicoCalc)

- P1. MMbg PiCa auf einen Blick
- P2. Voraussetzungen und Einrichten der Firmware
- P3. Installation und Start
- P4. Bildschirm, Terminal und Bedienung
- P5. Unterschiede zum PC3
- P6. Netze und Speicher auf dem PicoCalc
- P7. Die 16-Bit-Netze: Rundung und Spielstärke
- P8. Dateien
- P9. Hinweise und Grenzen

Lizenz und Dank

Neu seit dem letzten Release (1.10.34)

- **MMbg PC1:** neue Fassung für HDMI- und VGA-Boards ohne PSRAM, mit Oberfläche und Maus des PC3; die großen Netze liegen im **MODBUFF** (Flash, Kapitel 16).
- **PicoCalc:** Die großen Netze kommen jetzt ebenfalls aus dem MODBUFF. Zugliste, Undo und `.mat` sind so groß wie beim PC3, die PiCa **spielt gleich stark** (Teil 2, P6).
- **Bildschirmfotos** über die Firmware: **Druck** oder **Strg+** auf allen Geräten, auch als PNG (Kapitel 7.7).
- **Undo per Mausklick** fragt nach (6.1); Matchende „O wins the match 5-2“; Matchzähler auch im Terminal; kurze Eingabezeilen auf dem PicoCalc.

Teil 1 – MMbg PC3, Colour Maximite 2, PC1 und die gemeinsamen Grundlagen

1. Was ist MMbg?

MMbg ist ein vollständiges Backgammon-Programm in MMBasic. Es spielt **Matches** von 1 bis 15 Punkten, mit Doppler, Crawford-Regel und allem, was dazugehört.

- **Spielstärke:** Mit der Engine *big* spielt MMbg etwa auf dem Niveau von GNU Backgammon „Expert“ (0-ply). Gegen gnubg „World class“ (2-ply) gewinnt es rund **47 % der 5-Punkte-Matches** (577 von 1236 gemessenen Matches).
- **Grafik und Terminal:** Das Brett erscheint farbig (HDMI oder VGA 640 × 480, PicoCalc: LCD 320 × 320) mit animierten Zügen. Gleichzeitig zeigt ein serielles Terminal ein ASCII-Brett mit Zugliste, Bewertungen und Log.
- **Bedienung:** Gespielt wird mit der Tastatur oder, außer auf dem PicoCalc, mit der Maus. Auf Wunsch gibt es Geräusche für Würfeln und Ziehen.
- **Komfort:** Mehrstufiges Undo, automatisches Speichern und Fortsetzen. Matches werden als `.mat`-Datei für die Analyse in gnubg aufgezeichnet. Positionen lassen sich über **GNUbg IDs** austauschen.
- **Für Tüftler:** Über WLAN kann gnubg auf dem PC als Gegner dienen. Dazu kommen Testläufe über Nacht, Duplicate-Matches und ausführliche Logs.

2. Voraussetzungen

Welche Fassung für welches Gerät?

Gerät	Fassung	getestet mit
Pico Computer 3 (PC3), RP2350B mit PSRAM	`mmbg_pc3.bas`	PicoMitePC3 V7.00.00RC6
Colour Maximite 2 (Generation 2)	`mmbg_pc3.bas`	CMM2 7.00.00b4 (Kapitel 15)
PicoMite HDMI/USB-Board („PC1“), ohne PSRAM	`mmbg_pc1.bas`	PicoMiteHDMI V7.00.00RC6 (Kapitel 16)
Pico 2 mit VGA-Referenzboard	`mmbg_pc1.bas`	PicoMiteVGA V7.00.00RC6 (Kapitel 16)
PicoCalc mit Pico 2 W oder Pico 2	`mmbg_pica.bas`	WebMite V7.00.00RC6 (Teil 2)

Faustregel: Ein Board **mit PSRAM** nimmt `mmbg_pc3.bas`, ein HDMI- oder VGA-Board **ohne PSRAM** nimmt `mmbg_pc1.bas`.

Die Tabelle unten gilt für den PC3. Was auf den anderen Geräten anders ist, steht in Kapitel 15, Kapitel 16 und Teil 2.

Was	Anforderung
Rechner	**Pico Computer 3 (PC3)**: RP2350B mit PSRAM, HDMI, USB
Firmware	**PicoMitePC3 V7.00.00** (getestet mit RC6)

Was	Anforderung
Bildschirm	HDMI, MODE 3 (640 × 480, 16 Farben)
Tastatur	USB-Tastatur am PC3 und/oder serielles Terminal
Terminal (empfohlen)	VT100-fähig (z. B. Tera Term, PuTTY), **80 × 40 Zeichen**, Farben an
Maus (optional)	USB-Maus; eine PS/2-Maus wird ebenfalls gefunden (Kapitel 15)
Ton (optional)	`OPTION AUDIO` eingerichtet (beim PC3 I²S)
WLAN (optional)	nur für gnubg als Gegner (Kapitel 10)
SD-Karte (empfohlen)	Laufwerk B: für den Programmordner, Logs und `.mat`-Dateien

Auf dem PC3 liegen die großen Netze als Gleitkommazahlen im PSRAM (~512 KB), die Animation braucht weitere ~150 KB für einen Bildpuffer.

3. Installation

3.1 Dateien

Alle Dateien gehören **in einen Ordner** zusammen mit dem Programm (Kapitel 3.2). Das Paket enthält die Dateien für alle Ausgaben; jede Ausgabe übergeht einfach die Dateien, die sie nicht braucht (Tabelle in Kapitel 14 und in Teil 2, Kapitel P8).

Datei	Größe	Zweck	Fehlt sie …
`mmbg_pc3.bas`	~190 KB	das Programm für PC3 und CMM2	–
`mmbg_pc1.bas`	~185 KB	das Programm für HDMI/VGA ohne PSRAM (Kapitel 16)	–
`mmbg_pica.bas`	~195 KB	das Programm für den PicoCalc (Teil 2)	–
`bgnet345.dat`	~96 KB	kleine Netze 3, 4, 5 (gnubg)	nur Engines 0–2
`bgnet02.dat`	Text	Ausgabe-Gewichte der großen Netze	keine Engines 4 und 5
`bgnet0.bin`, `bgnet2.bin`	je 256 000 Byte	große Netze Kontakt und crashed (PC3, CMM2)	keine Engines 4 und 5
`bgesc.bin`	64 KB	Escape-Tabellen (gnubg; PC3, CMM2)	keine Engines 4 und 5
`bgnet16.mb`, `bgpk16.dat`	144 192 Byte, Text	große Netze mit 16 Bit und Escape-Tabellen für den MODBUFF, Prüfsummen (PC1, PicoCalc)	keine Engines 4 und 5
`bgbo15.dat`	~1,6 MB	Bear-off-Datenbank bis 15 Steine	Rennen nur mit Netz 5
`bgnet35.dat`, `bgbo10.dat`		kleinere Ersatzdateien, nur benutzt, wenn `bgnet345.dat` oder `bgbo15.dat` fehlt	–

Die Einstellungs- und Spielstandsdateien (`mmbg_pc3.ini`, `bgext.ini`, `mmbg_pc3.sav`; beim PC1 `mmbg_pc1.…`, beim PicoCalc `mmbg_pica.…`) legt MMbg selbst im Programmordner an, dazu den Ordner `log/` darunter.

3.2 Alles in einem Ordner auf der SD-Karte

MMbg sucht alle seine Dateien **relativ zum Programmordner**: Netze, Datenbank, Einstellungen, gespeichertes Match und den Log-Ordner darunter. Nichts ist fest auf ein Laufwerk eingestellt. Der Ordner kann deshalb **beliebig heißen und auf jedem Laufwerk in jeder Tiefe liegen** – das ist auch der Grund, warum dasselbe Paket auf allen Rechnern läuft, ob die SD-Karte dort `B:` heißt (PC3, PicoCalc) oder `A:` (Colour Maximite 2).

Empfohlen wird:

Ordner	Inhalt
`/mmbg`	Programm, Netze, Datenbank, ini-Dateien, gespeichertes Match
`/mmbg/log`	Logs und `.mat`-Dateien (legt MMbg selbst an)

Dazu das Paket entpacken und den Ordner `mmbg` in die Wurzel der SD-Karte kopieren. Alternativ jede Datei mit `XMODEM RECEIVE "name"` in einen Ordner übertragen, den man vorher mit `MKDIR "B:/mmbg"` angelegt hat.

3.3 Starten

Gestartet wird immer **aus dem Programmordner**, damit MMbg seine Dateien dort findet:

```
CHDIR "B:/mmbg"
LOAD "mmbg_pc3.bas"          (PC1, VGA-Board: "mmbg_pc1.bas")
RUN
```

Gerät	Start
PC3, PC1, VGA-Board	`LOAD "mmbg_pc3.bas"` bzw. `"mmbg_pc1.bas"`, dann `RUN`. Die veröffentlichten Programme haben nur kurze Kommentare und laden ohne `, C`. Mit `, C` lässt MMBasic die Kommentare weg und es bleibt etwas mehr Programmspeicher.
PicoCalc	**`LOAD "mmbg_pica.bas", C`**, dann `RUN`. Hier ist `, C` nötig, sonst reicht der Speicher nicht (Teil 2).
Colour Maximite 2	`CHDIR "mmbg"`, dann `RUN "mmbg_pc3.bas"`. Die SD-Karte heißt dort `A:` (Kapitel 15).

Beim Start erscheinen Logo, Lizenz und die Ladezeilen für Netze, Datenbank und WLAN. Danach wartet MMbg im Startbild auf **N** (neues Match).



Bild 1: Das Terminal nach dem Start: ASCII-Brett, darunter die Ladezeilen für Netze und Datenbank.

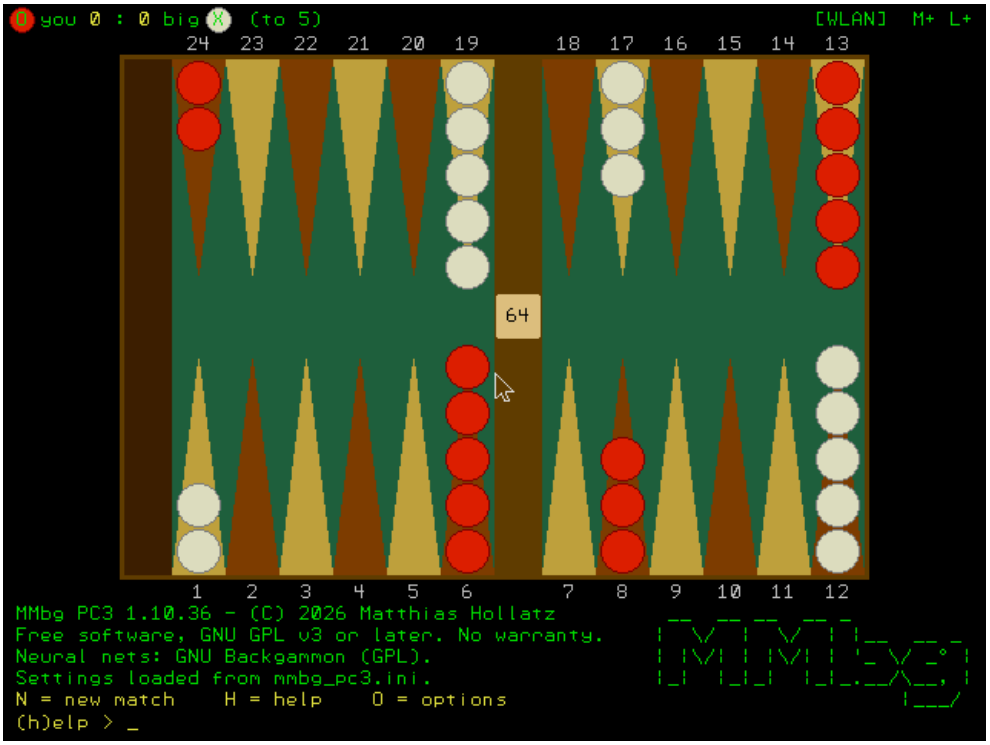


Bild 2: Startbild auf dem HDMI-Bildschirm. Unten links Version, Lizenz und Meldungen, rechts das Logo, darunter die wichtigsten Tasten.

> **Fehlermeldungen mit Zeilennummer** beziehen sich beim Laden mit `, C` auf das Programm ohne Kommentarzeilen.

3.4 Der Log-Ordner

Logs und `.mat`-Dateien kommen in den **Log-Ordner** (Bildschirmfotos dagegen in die Wurzel der SD-Karte, Kapitel 7.7). Standard ist `log/` unter dem Programmordner (`B:/mmbg/log`); MMbg legt ihn beim Start an, wenn es ihn nicht gibt. Einzurichten ist nichts.

Für einen anderen Ordner die Optionsseite öffnen (**O → 7**), **L** drücken und den Ordner eingeben:

Eingabe	Log-Ordner
`log`	`log/` unter dem Programmordner (Standard)
`B:/spiele/bglog`	jeder vorhandene Ordner, absolut
`-`	der Programmordner selbst

Mit **S** speichern. In `bgext.ini` steht die Einstellung in der Zeile `logdir=`: leer heißt `log/`, `.` heißt Programmordner.

4. Der Bildschirm und das Terminal

4.1 HDMI-Bildschirm

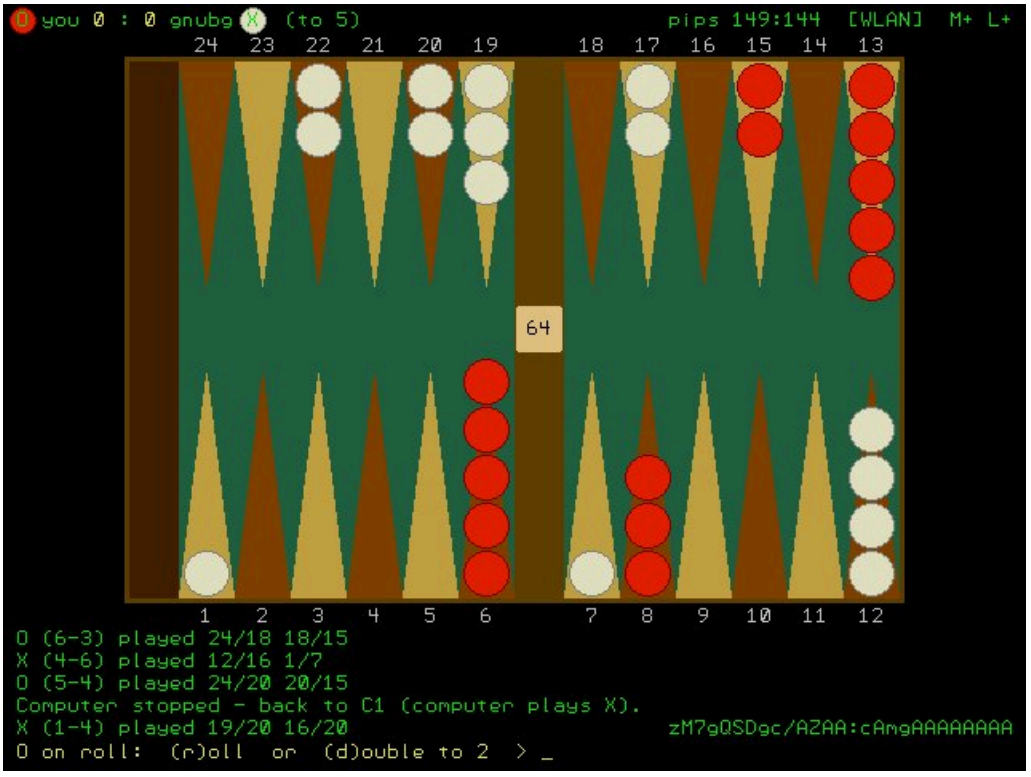


Bild 3: Laufendes Spiel. Titel mit Spielern, Stand, Pips und Schaltern, darunter das Brett mit dem Doppler in der Bar, das Log mit fünf Zeilen, rechts unten die GNUbg ID und ganz unten der Prompt (hier die Frage vor dem Wurf).

- **Titelzeile**
  - O und X erscheinen als kleine Steine in der Spielerfarbe, daneben der Spieler: `you` für einen Menschen, sonst der Name der Engine.
  - Der **Spielstand ist gelb**, dahinter steht die Matchlänge, z. B. `(to 5)`.
  - `M 4:2` zählt die gewonnenen Matches der Sitzung. Er erscheint bei Computerläufen und sobald ein Match zu Ende ist.
  - In der Mitte steht `Crawford` oder `Post-Crawford`.
  - Rechts folgen die Pips (Option), `[WLAN]`, `M±` für die `.mat`-Aufzeichnung und `L±` für das Log.

- **Brett:** 24 Punkte, Bar, Ablage. Die Würfel liegen immer **rechts vom Spieler** aus gesehen, also O-Würfel rechts und X-Würfel links.
  - Der Doppler steht in der Mitte der Bar oder beim Besitzer. Im Crawford-Spiel ist er grau mit „64“; Doppeln ist dann verboten.
  - Ein **angebotenes Doppel** erscheint als einzelner Doppler im Spielfeld auf der Seite dessen, der entscheiden muss.
- **Log:** Die letzten fünf Meldungen, Züge und Hinweise stehen über dem Prompt.
- **GNUbg ID:** Die ID der aktuellen Stellung steht grün rechts in der untersten Log-Zeile (Kapitel 7.4).
- **Farben:** Text ist grün, Tasten, Eingaben und veränderliche Werte sind gelb.

Auf dem Colour Maximite 2 und mit MMbg PC1 (HDMI oder VGA) sieht der Bildschirm genauso aus.

4.2 Terminal

Das Terminal (80 × 40, VT100) zeigt dasselbe Spiel als Text:

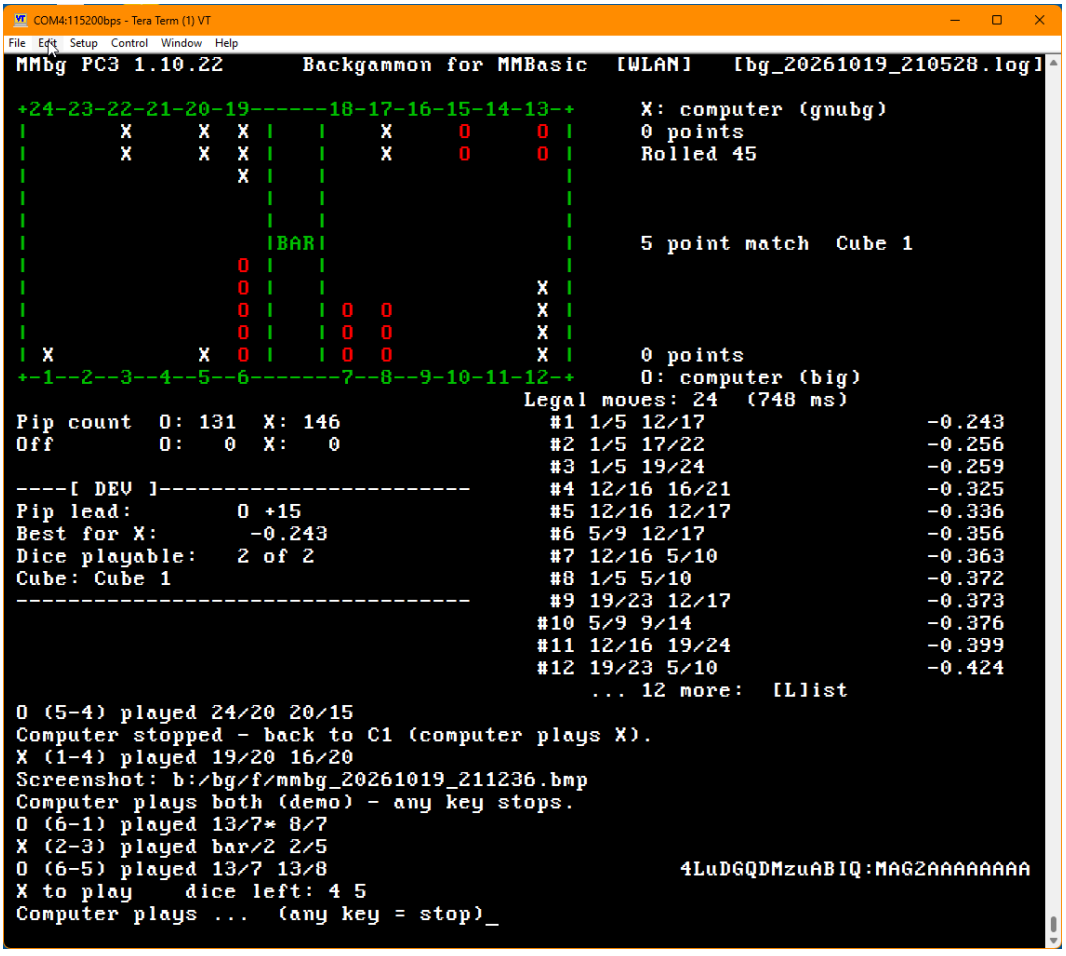


Bild 4: Das Terminal während eines Computerlaufs (big gegen gnubg): ASCII-Brett, rechts Spieler und Würfel, darunter Pips, DEV-Spalte, Zugliste mit Bewertungen und das Log mit der GNUbg ID.

- **ASCII-Brett:** Der Rahmen ist grün, O-Steine sind weiß, X-Steine rot. Die Option *swap* tauscht die Farben, die Option *flip* spiegelt das Brett wie in der Grafik.
- **Rechts neben dem Brett:** Spieler, Punkte, Matchlänge, Doppler und die gewürfelten Augen.
- **Darunter:** Pips, abgetragene Steine und die Bar. Die **DEV-Spalte** (Taste `P`) zeigt Pip-Vorsprung, den besten Wert, die spielbaren Würfel und den Doppler.

- **Zugliste rechts:** die 12 besten Züge mit Bewertung (Equity), den Rest mit `L`. Vor dem Wurf erscheint stattdessen die Doppler-Analyse: Gewinnchancen im Match bei No double, Double/take und Double/pass sowie die Empfehlung des Computers.
- **Unten:** die letzten **8 Log-Zeilen** und die **GNUbg ID** rechts. Von dort lässt sie sich bequem mit der Maus kopieren.

5. Ein Match spielen

5.1 Start

- **N** startet ein neues Match mit der eingestellten Länge.
- **M7** startet ein Match bis 7 Punkte. Möglich sind 1 bis 15 Punkte.
- **C1** (Standard): Der Mensch spielt O (unten), der Computer spielt X (oben).
- Der Eröffnungswurf bestimmt, wer beginnt.

5.2 Ziehen

Eingabe	Bedeutung
`13/7 8/7`	Züge in gnubg-Schreibweise, mehrere Teilzüge durch Leerzeichen getrennt
`bar/22`, `6/off`	von der Bar, abtragen (kurz: `b`, `o`; auch `-` statt `/`)
`13 24`	nur die Startpunkte: der große Würfel zuerst
`b 6`	von der Bar und von der 6
`#3`	Zug Nr. 3 der Zugliste
`E`	Zug beenden, falls kein Würfel mehr spielbar ist
`U`	Undo (Kapitel 6)

Ein Teilzug ist erst gültig, wenn danach noch die größtmögliche Zahl Würfel spielbar bleibt. Kann ein Spieler gar nicht ziehen, passt MMbg automatisch und zeigt die Würfel kurz an.

Mit der Maus (PC3, CMM2, PC1):

Aktion	Bedeutung
Links auf einen eigenen Stein (oder die Bar)	mit dem großen Würfel ziehen. Geht das nicht, mit dem kleinen
Rechts auf einen eigenen Stein (oder die Bar)	mit dem kleinen Würfel ziehen. Geht das nicht, mit dem großen
Klick auf die eigenen Würfel	Undo, nach Rückfrage (Kapitel 6.1)
Vor dem Wurf: Klick auf den Doppler	doppeln
Vor dem Wurf: Klick aufs Brett	würfeln
Klick auf den angebotenen Doppler	annehmen. Aufgeben nur mit der Taste `P`

Beispiel: Ein Stein steht auf der Bar, gewürfelt ist 3-1. Ein **Rechtsklick** auf die Bar setzt mit der 1 ein, die 3 bleibt für einen anderen Stein.

5.3 Doppler

Vor jedem Wurf fragt MMbg den Spieler am Wurf, sofern er doppeln darf: `(r)oll or (d)ouble`. Bietet der Gegner ein Doppel an, lautet die Antwort `(t)ake` oder `(p)ass`.

- Mit der Option *hint* zeigt das Log die Doppler-Empfehlung und die Match-Gewinnchancen.
- Im **Crawford-Spiel** ist Doppeln verboten. Danach folgt Post-Crawford.
- Gespielt wird nach Match-Regeln. Geldspiel-Regeln wie Jacoby oder Beaver gibt es nicht.

5.4 Spiel- und Matchende

Gewinnt ein Spieler, zählen ein Punkt (einfach), zwei (Gammon) oder drei (Backgammon), jeweils mal Doppler. Ein Spiel bringt also 1, 2, 3, 4, 6, 8, 12 … Punkte. Das nächste Spiel beginnt nach drei Sekunden.

Ist das Match entschieden, meldet MMbg zum Beispiel `*** 0 wins the match 5-2 ***`. Ein neues Match startet mit ENTER, bei Computerläufen nach drei Sekunden von selbst.

6. Undo, Speichern und Fortsetzen

6.1 Undo

- **U** im laufenden Zug nimmt die eigenen Teilzüge zurück.
- **U** ohne gezogenen Stein springt zum **letzten eigenen Zuganfang** zurück, mit denselben Würfeln. Das geht auch, wenn der Computer schon gezogen hat, zum Beispiel nach einem Verklicken.
- **Weiteres U** geht noch einen eigenen Zug zurück und so weiter, bis zum Anfang des Spiels.
- Nach einem Undo würfelt der Computer neu, damit niemand auf dessen Würfel reagieren kann. Undo wirkt nur innerhalb des laufenden Spiels.
- **Mit der Maus:** Ein Klick auf die eigenen Würfel fragt erst nach: `Undo? click the dice again or (y)es / (n)o`. Ein zweiter Klick auf die Würfel oder **Y** nimmt zurück, alles andere nicht. So kostet ein Verklicken keinen Zug. Die Taste **U** nimmt ohne Rückfrage zurück.

Grundlage ist der **Zugverlauf**: Zu jedem Zuganfang merkt sich MMbg die GNUbg ID, also Brett, Stand, Doppler und Würfel. Der PC3 und der PicoCalc behalten 300 Zuganfänge, MMbg PC1 die letzten 40.

6.2 Speichern und Fortsetzen

- MMbg speichert das laufende Match **automatisch** in `mmbg_pc3.sav` (PC1: `mmbg_pc1.sav`, PicoCalc: `mmbg_pica.sav`):
  - bei jedem Zuganfang eines Menschen,
  - an den Fragen roll/double und take/pass,
  - bei Computerläufen C2 und C3 in jedem Zug. C4 speichert nicht.
- **SG** speichert sofort, **LG** lädt das gespeicherte Match.
- Beim **Start** fragt MMbg: *„Saved match found – continue it? (y/n)“*.
- Gespeichert werden der ganze Zugverlauf (Undo geht nach dem Laden weiter), Computer-Modus, Engines und die laufende `.mat`-Aufzeichnung.
- Am Matchende wird die Datei gelöscht.

Was in der Datei steht: Die Datei ist reiner Text, eine Angabe je Zeile. Das Herzstück sind die `turn=`-Zeilen: Für **jeden Zuganfang** des Spiels steht dort die **GNUbg ID** der Stellung (Brett, Stand, Doppler, Würfel), dazu die Zugnummer und wie viele `.mat`-Einträge bis dahin geschrieben waren. Aus der letzten ID baut MMbg beim Laden die Stellung wieder auf; die früheren IDs machen Undo nach dem Laden möglich.

```
' MMbg PC3 saved match
version=1.10.36
date=25-10-2026 20:26:53
computer=1                    Computer-Modus (C1)
engine_o=0                    Engines O und X
engine_x=4
game=7                        Spielnummer der Sitzung
turn=4HPwATDgc/ABMA:cAmgAAAAAAAA 0 0     Zuganfang: GNUbg ID, Zug, .mat-Stand
turn=4HPwATDgc/ABMA:QQnrABAAGAAA 1 1
...
matfile=log/mmbg_20261025_202011.mat
mgame=3                       Spielnummer im Match (.mat)
act=2|52: 13/8 24/22          bisher aufgezeichnete Aktionen des Spiels
now=4HPwATDgc/ABMA:cAmgAAAAAAAA          Stellung vor dem Wurf (an roll/double)
```

Jede ID lässt sich auch einzeln in gnubg einfügen, um eine Stellung aus einem gespeicherten Match anzusehen. Die Erklärungen rechts stehen nur hier im Handbuch, nicht in der Datei.

7. Aufzeichnung, Spielanalyse und Bildschirmfotos

7.1 Matches als `.mat` für gnubg

Mit der Option **mat** (`M+` im Titel) schreibt MMbg jedes Match in eine eigene Datei:

```
[Log-Ordner]mmbg_JJJJMMTT_HHMMSS.mat
```

- Das Format ist dasselbe, das gnubg mit `export match mat` schreibt.
- Der Kopf wird beim Matchstart geschrieben, jedes Spiel am Spielende angehängt.
- Neue Matches überschreiben alte Dateien nie.
- Was mit Undo zurückgenommen wurde, erscheint nicht in der Datei.
- Ein fortgesetztes Match schreibt in seine alte Datei weiter.
- X steht in gnubg als Spieler 0 (oben), O als Spieler 1 (unten).

Analyse in gnubg:

1. Die Datei zum PC übertragen, z. B. per SD-Karte oder XMODEM. XMODEM füllt das Dateiende mit Nullbytes auf, das stört gnubg nicht.
2. In gnubg: *Datei → Importieren → .mat* oder `import mat datei.mat`.
3. *Analyse → Match analysieren* oder `analyse match`, dann die Statistik mit `show statistics match`.

7.2 Log-Datei

Mit **G** an oder aus (`L±` im Titel). Die Einstellung wird mit den übrigen Optionen gespeichert. Die Datei `bg_JJJJMMTT_HHMMSS.log` im Log-Ordner enthält:

- jeden Zug mit Zeit, Würfeln, Platz in der Zugliste und Bewertung,
- die Rechenzeit, die Pips und die Antwortzeit von gnubg,
- Doppler-Entscheidungen mit Gewinnchancen sowie Spiel- und Matchergebnisse.

Ab etwa 1 MB beginnt eine neue Datei. Mit **GN** beginnt sofort eine neue.

Eine Zugzeile sieht so aus (im Log steht sie in einer Zeile):

```
  2105.3s G12 T 14 X 6-4 big   24/18 13/9          #1/18 +0.123 best +0.123
         gen 512ms  pip O 140 X 150  ext 70ms
```

Das heißt: Sekunden seit dem Start, Spiel 12, Zug 14, X hat 6-4 gewürfelt, Engine big, der gespielte Zug, sein **Platz in der Zugliste** (1 von 18 legalen Zügen), seine Bewertung und die beste Bewertung, die Rechenzeit des Zuggenerators, die Pips beider Seiten und bei gnubg die Antwortzeit. Dazu kommen Zeilen mit `---` (Doppler, Engines, Einstellungen, Undo), `***` (Spielende), `###` (Matchende und Gesamtstand), `!!!` (Fehler) und, auf dem PicoCalc, `D` (Brett-Dump mit Taste D).

7.3 `.mat` oder Log – was wofür?

	`.mat` -Aufzeichnung	Log-Datei
Zweck	**das Match nachspielen und analysieren**	**die Engines beobachten**, vor allem bei Testläufen
Format	Standardformat, liest gnubg (und andere Programme)	eigenes Textformat für Mensch und Auswerteskript
Inhalt	nur das Spiel: Würfe, Züge, Doppler, Ergebnisse	zusätzlich Platz in der Zugliste, Bewertungen, Rechenzeiten, Pips, gnubg-Antwortzeiten, Fehler, Einstellungen
Umfang	eine Datei je Match, klein	eine laufende Datei, ab ca. 1 MB eine neue
typisch	eigene Partien gegen den Computer	Long Runs über Nacht (Kapitel 11)

Beides kann gleichzeitig an sein. Wer nur spielt und seine Partien in gnubg analysieren will, braucht nur `.mat`.

7.4 GNUbg ID

Die ID beschreibt eine Stellung vollständig: Brett, Matchlänge, Stand, Doppler, Crawford, Spieler am Wurf und Würfel. Ein Beispiel:

```
4HPwATDgc/ABMA:cAmgAAAAAAAA
```

- **MMbg → gnubg:** Die ID steht rechts unten und auf der Dump-Seite (Taste **D**). Im Terminal kopieren und in gnubg in das Feld *GNUbg ID* einfügen oder `set gnubgid …` eingeben. So lässt sich jede Stellung sofort in gnubg untersuchen.
- **gnubg → MMbg:** Eine ID einfach am Prompt einfügen, auch an roll/double oder take/pass. MMbg fragt *„Take over the position…? (y/n)“* und baut die Stellung auf. Ein Geldspiel, ein Match über 15 Punkte und ein Doppler über 64 werden abgelehnt.

- Die IDs sind mit gnubg 1.08.003 gegengeprüft. Ein angebotenes Doppel enthält die ID nicht, weil gnubg es beim Einlesen verwirft.

7.5 Tipp-Option

Mit **hint** (Option 3) schreibt MMbg den besten Zug ins Log, wenn ein Mensch am Zug ist, und an den Doppler-Fragen die Empfehlung. Das eignet sich gut zum Lernen.

7.6 Brett-Dump (D)

D zeigt die Stellung als Text: Steine je Punkt, Bar, abgetragene Steine, die gewürfelten Augen, die noch zu spielenden Würfel und die **GNUbg ID**. So lässt sich eine Stellung am einfachsten notieren oder melden, zum Beispiel zusammen mit einem Fehlerbericht. Auf dem PicoCalc stehen Brett und ID zusätzlich im Log (Zeilen, die mit `D` beginnen).

7.7 Bildschirmfoto (Druck oder Strg+\)

MMbg schaltet beim Start `OPTION PRINTSCREEN ON` ein. Damit speichert die **Firmware** den Bildschirm, sobald die Taste **Druck** (Print Screen) oder **Strg+** gedrückt wird, auch mitten in einer Animation. Das geht auf allen Geräten: PC3, Colour Maximite 2, PC1, VGA-Board und PicoCalc (dort das LCD mit 320 × 320).

Die Datei landet in der **Wurzel der SD-Karte**, nicht im Log-Ordner:

```
SCREEN_JJJJMMTT_HHMMSS.BMP
```

Als PNG: Einmalig am Prompt eingeben (die Einstellung bleibt gespeichert):

```
OPTION PRINTSCREEN FORMAT PNG
```

Dann entstehen PNG-Dateien, viel kleiner als BMP. Ab PicoMite **V7.00.00RC6** bzw. CMM2 **7.00.00b4** zeigt der Bildbetrachter des Dateimanagers (`FM`) die PNG-Bilder direkt auf dem Gerät. Die Bilder in diesem Handbuch sind so entstanden.

Ältere Firmware ohne `OPTION PRINTSCREEN` überspringt den Befehl beim Start, dann gibt es eben keine Bildschirmfotos. Das eigene **Strg+P** früherer Versionen gibt es nicht mehr.

8. Engines und Spielstärke

8.1 Die Engines

Nr.	Name	Arbeitsweise	Stärke (grob)
0	**base**	Bewertungsformel: Pips, Blots, Punkte, Bar	Anfänger
1	**book**	base plus Eröffnungsbuch	Anfänger+
2	**tuned**	im Labor abgestimmte Gewichte	Gelegenheitsspieler
3	**net**	kleine gnubg-Netze (0-ply)	guter Klubspieler
4	**big**	große gnubg-Netze, Vorfilter mit den kleinen Netzen	**≈ gnubg „Expert“ (0-ply)**

Nr.	Name	Arbeitsweise	Stärke (grob)
5	**deep**	big mit 1 Wurf Vorausschau (1-ply)	stärker als big, ca. 18-mal langsamer
E	**gnubg**	gnubg auf dem PC über WLAN	so stark wie gnubg eingestellt ist

Gewählt wird mit **A45**: Engine für O ist 4, für X ist 5. Für jede Seite gibt es ein Zeichen aus 0 bis 5 oder E.

Messwerte für big:

Messung	Ergebnis
Matches gegen gnubg 2-ply (bis 5), PC3	**577 von 1236 = 46,7 %**
Matches gegen gnubg 2-ply (bis 5), PicoCalc (16 Bit)	199 von 450 = 44,2 % ± 4,6 % (Teil 2, P7)
Fehler je Zug, gemessen an gnubg 2-ply	big 2,6 mEq, gnubg 0-ply 2,4 mEq
bester Zug getroffen	big 82 %, gnubg 0-ply 83 %

- Rund 90 % des Rückstands auf gnubg 2-ply kommen aus der Zugwahl, rund 10 % aus dem Doppler.
- Die Rechenzeit liegt auf allen Rechnern im Mittel bei etwa 0,4 bis 0,6 s je Zug, mit *deep* bei etwa 10 s.

8.2 Doppler-Modell

Für den Doppler rechnet MMbg wie gnubg mit 0-ply: Aus den Gewinn-, Gammon- und Backgammon-Chancen des Netzes entstehen über die **Match-Gewinntabelle Kazaross-XG2** die Chancen für No double, Double/take und Double/pass.

8.3 Computer-Modi

Befehl	Modus
`C0`	zwei Menschen
`C1`	Computer spielt X (Standard)
`C2`	Computer spielt beide Seiten (zum Zuschauen, mit Pausen)
`C3`	beide, ohne Pausen und ohne Animation
`C4`	**Turbo**: beide, ohne Brettanzeige, endlos Matches (Testläufe)

In C2 bis C4 stoppt **jede Taste** den Computer.

9. Die neuronalen Netze

9.1 Wie arbeitet so ein Netz? (ganz einfach erklärt)

Ein neuronales Netz ist im Kern eine **große Rechenvorschrift mit vielen festen Zahlen**. Es bekommt eine Stellung und gibt zurück, wie gut sie ist. Mehr nicht – es denkt nicht voraus und

kennt keine Regeln.

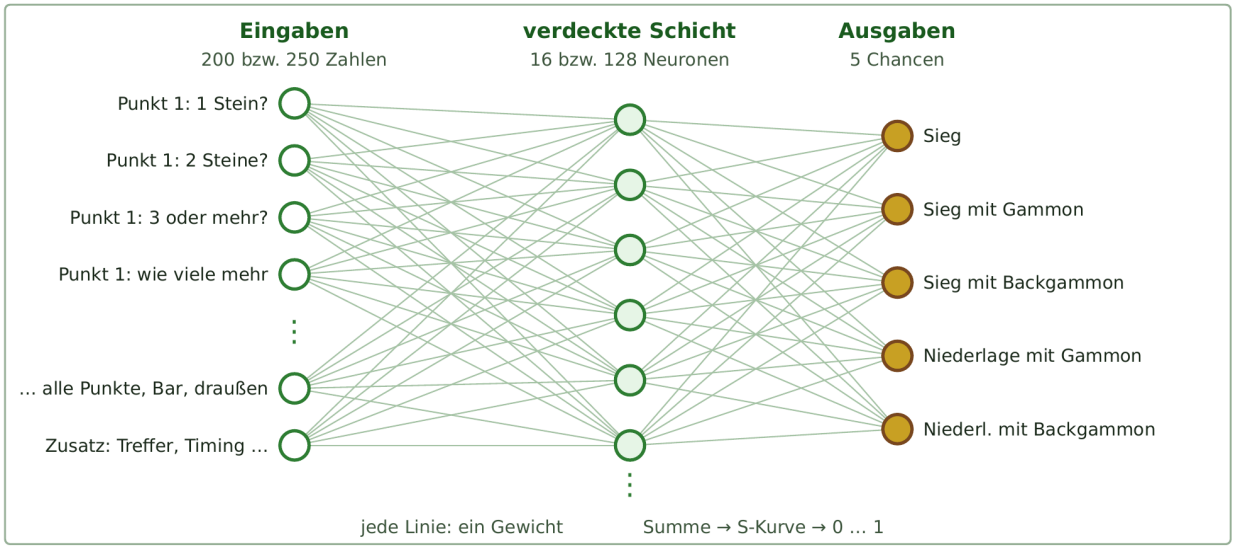


Bild 5: Aufbau eines gnubg-Netzes (verkleinert dargestellt). Links die Stellung als Zahlen, in der Mitte die verdeckte Schicht, rechts die fünf Chancen.

1. Die Stellung wird zu Zahlen. Für jeden Punkt und jede Seite gibt es vier Eingaben: Liegt dort genau 1 Stein (ein Blot)? Genau 2? 3 oder mehr? Und wie viele darüber hinaus? Dazu kommen Bar und abgetragene Steine. So entstehen 200 Zahlen, meist 0 oder 1. Das große Netz bekommt noch 50 Zusatzmerkmale, die ein Mensch „sehen“ würde: Wie oft kann ich getroffen werden? Wie leicht kommen hintere Steine heraus? Wie viel Zeit bleibt mir?

2. Die verdeckte Schicht. Jedes Neuron in der Mitte bildet eine **gewichtete Summe** aller Eingaben: jede Eingabe mal ihr eigenes Gewicht, alles zusammengezählt. Die Summe läuft durch eine **S-Kurve**, die sie auf einen Wert zwischen 0 und 1 drückt. Jedes Neuron wird so zu einem kleinen „Detektor“ für ein Muster, etwa „gutes Heimfeld“ oder „gefährdeter Blot“ – welches Muster, hat das Netz selbst gelernt.

3. Die Ausgaben. Die fünf Ausgabe-Neuronen machen dasselbe noch einmal mit den Werten der Mitte. Heraus kommen fünf Chancen zwischen 0 und 1: Sieg, Sieg mit Gammon, Sieg mit Backgammon, Niederlage mit Gammon, Niederlage mit Backgammon. Daraus wird ein einziger Wert, die **Equity**:

```
Equity = 2 x Sieg - 1 + Gammon-Siege + Backgammon-Siege - Gammon-Niederlagen -
Backgammon-Niederlagen
```

4. Woher kommen die Gewichte? Sie sind das eigentliche Wissen. Die gnubg-Autoren haben sie mit sehr vielen ausgespielten Stellungen trainiert: Das Netz schätzt, vergleicht mit dem tatsächlichen Ergebnis und verschiebt die Gewichte ein klein wenig in die richtige Richtung – millionenfach. MMbg rechnet nur mit den fertigen Zahlen, es lernt nicht selbst.

5. Wie wird daraus ein Zug? Der Zuggenerator findet alle legalen Züge (oft 20 bis 50, bei Paschen auch einige hundert). Für jeden Zug wird die **Stellung danach** durch das Netz geschickt. Gespielt wird der Zug mit der besten Equity. Engine *big* bewertet erst alle Züge mit dem kleinen Netz und rechnet nur die besten 12 mit dem großen nach; *deep* schaut zusätzlich auf alle 21 möglichen Würfe des Gegners voraus.

Das große Netz braucht für eine Stellung rund 32 000 Multiplikationen (250 × 128 + 128 × 5). In MMBasic dauert das auf dem PC3 etwa 19 ms. Da in einer echten Stellung die meisten der 250

Eingaben null sind, zählt MMbg nur die Zeilen der Eingaben zusammen, die nicht null sind (rund 50).

9.2 Die Netze in MMbg

MMbg verwendet die **Netzgewichte von GNU Backgammon** (gnubg 1.08, GPL). MMbg rechnet die Netze selbst in MMBasic. Die Eingabekodierung ist dieselbe wie in gnubg.

Netz	Aufbau	Einsatz
3 Kontakt (klein)	200 Eingaben, 16 verdeckte Neuronen, 5 Ausgaben	Engine net, Vorfilter für big
4 crashed (klein)	200-16-5	Stellungen mit zusammengebrochenem Heimfeld
5 Rennen (klein)	200-8-5	Rennen, wenn kein Kontakt mehr besteht
0 Kontakt (groß)	250 Eingaben (200 + 50 Zusatzmerkmale), 128 Neuronen, 5 Ausgaben	Engine big und deep
2 crashed (groß)	250-128-5	Engine big und deep

- **Die 5 Ausgaben** sind: Sieg, Sieg mit Gammon, Sieg mit Backgammon, Niederlage mit Gammon, Niederlage mit Backgammon.
- **Die 50 Zusatzmerkmale** des großen Netzes, etwa Treffer-Chancen, Escapes, Containment und Timing, berechnet MMbg nach dem Vorbild von gnubg.
- **Bear-off-Datenbank `bgbo15.dat`:** Sie ist selbst berechnet und enthält keine gnubg-Daten. Sie liefert exakte Werte im Endspiel: eine Seite, 6 Punkte, bis 15 Steine, 54 264 Stellungen, mit Gammon-Anteilen.
- **Auf PC1 und PicoCalc** liegen dieselben Gewichte mit 16 Bit je Gewicht im MODBUFF (Flash) – siehe Kapitel 16 und Teil 2, Kapitel P6 und P7.

Welche Netze welche Engine braucht:

Engine	benötigt
0 bis 2	nichts, nur das Programm
3 net	`bgnet345.dat`
4 big, 5 deep	zusätzlich `bgnet02.dat` und die großen Netze: PC3 und CMM2 `bgnet0.bin`, `bgnet2.bin`, `bgesc.bin`; PC1 und PicoCalc `bgnet16.mb`, `bgpk16.dat`
alle	`bgbo15.dat` macht das Endspiel exakt

10. Spielen gegen gnubg über WLAN

Der PC3 (und der PicoCalc mit Pico 2 W) kann **GNU Backgammon auf einem PC** im Heimnetz als Gegner nutzen. MMbg stellt dabei die Fragen, gnubg rechnet. Der Colour Maximite 2 und MMbg PC1 haben kein WLAN.

10.1 Einrichten

Schritt 1 – WLAN am PC3 einmalig einrichten und neu starten:

```
OPTION WIFI "MeinNetz", "Passwort"
```

Schritt 2 – gnubg auf dem PC starten (z. B. `gnubg-cli`) und eingeben:

```
external :41000
```

gnubg wartet dann auf Port 41000. Die Spielstärke wird in gnubg selbst eingestellt, z. B. 0-ply oder 2-ply. Das Paket enthält im Ordner `gnubg_start_PC` fertige Batch-Dateien für Windows (0-, 1- und 2-ply).

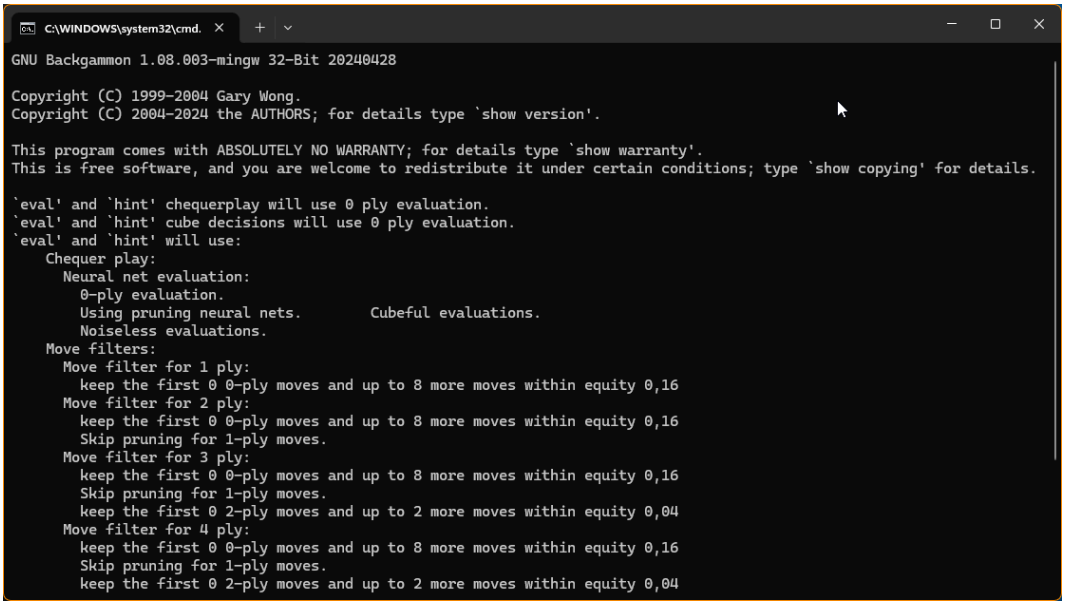


Bild 6: gnubg 1.08.003 auf dem PC (Kommandozeile), eingestellt auf 0-ply. Danach folgt `external :41000`.

Schritt 3 – in MMbg die Optionsseite öffnen (**O → 7**) und einstellen:

- **H** host: die IP-Adresse des PCs
- **P** port: z. B. 41000
- **W** timeout in Sekunden
- **F** fallback: die Engine, die einspringt, wenn gnubg nicht antwortet (z. B. 4 big)
- **T** testet die Verbindung (Version und 10 Pings)
- **S** speichert alles in die ini-Dateien

Steht `[WLAN]` im Titel, ist das WLAN da.

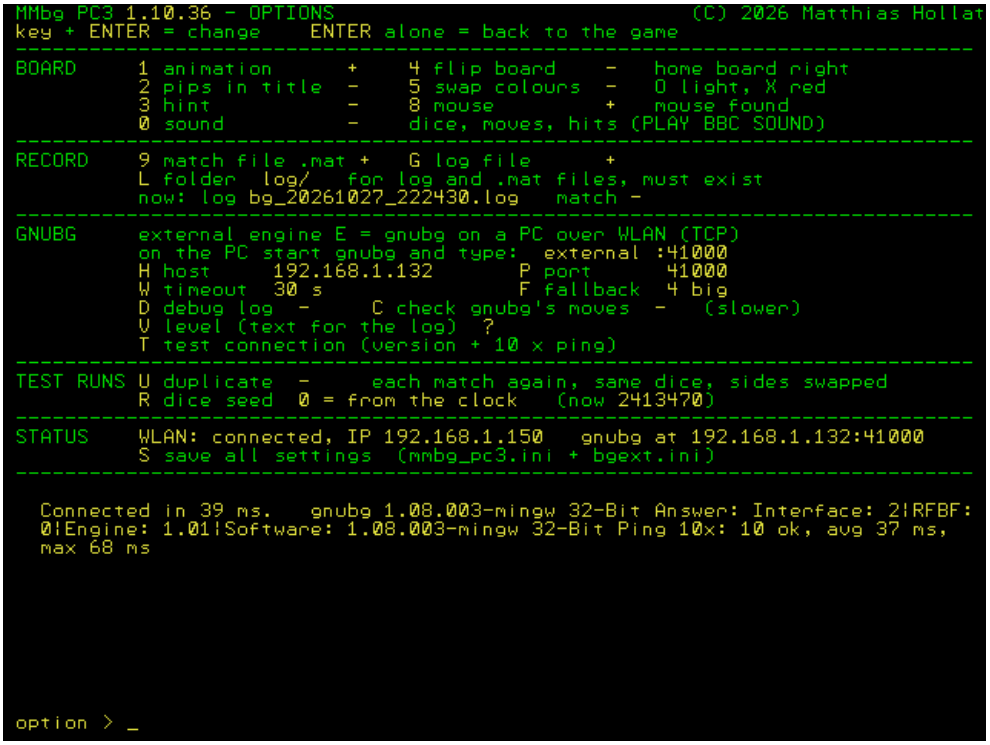


Bild 7: Verbindungstest mit T geglückt: unten die gnubg-Version und 10 Pings mit Antwortzeiten.

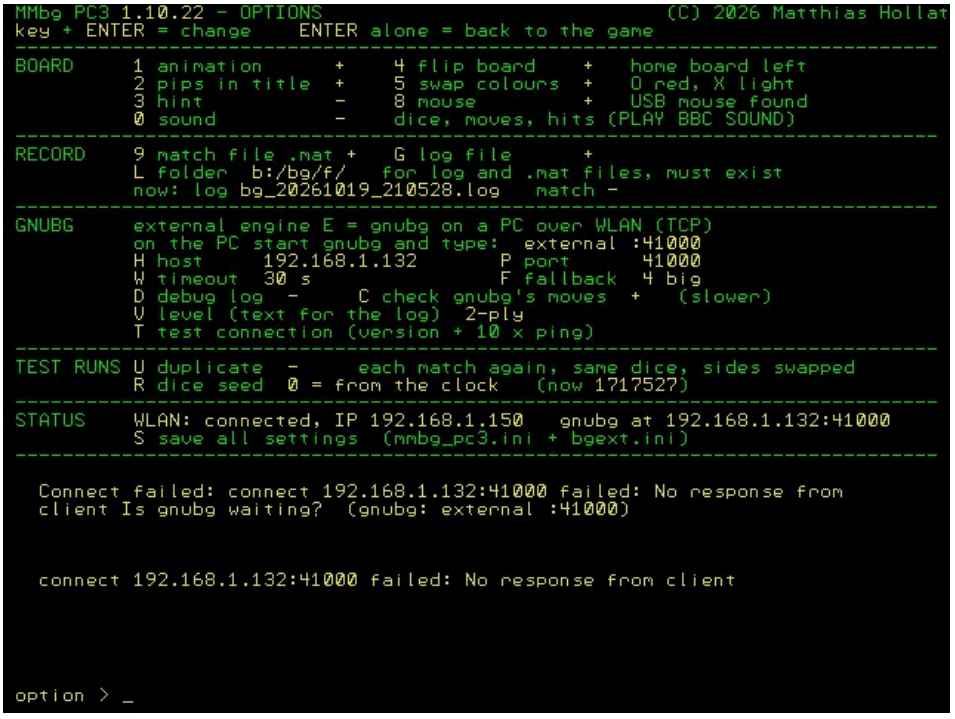


Bild 8: Verbindungstest gescheitert: gnubg läuft nicht oder wartet noch nicht (`external :41000` vergessen), oder Host/Port stimmen nicht.

10.2 Spielen

Befehl	Wirkung
`A4E`	O = big, X = gnubg
`C1`	der Mensch spielt O, gnubg spielt X: ein Spiel gegen gnubg
`C2` oder `C3`	big gegen gnubg zum Zuschauen

Befehl	Wirkung
`C4E`	Engines O 4, X gnubg und Turbo

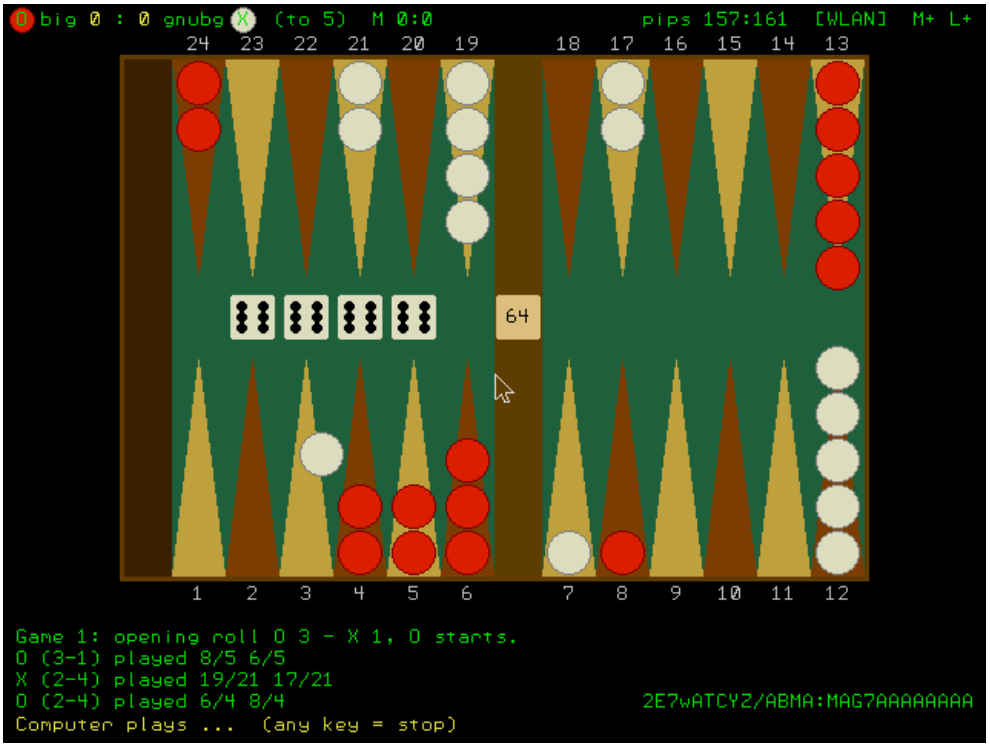


Bild 9: X ist gnubg. Das Log meldet Version, Stufe in Klammern (hier 2-ply), Adresse und die Ersatz-Engine (fallback big); rechts unten die GNUbg ID.

Weitere Schalter auf der Optionsseite:

- **D debug log:** Anfrage und Antwort von gnubg stehen im Log.
- **C check:** MMbg bewertet auch die gnubg-Züge mit. Das ist langsamer, aber gut für Vergleiche.
- **V level:** nur ein Text für das Log und die Meldung beim Start, z. B. „2-ply“. Ist nichts eingetragen, steht dort `(?)`, etwa `gnubg 1.08.003-mingw 32-Bit (?) at 192.168.1.132` – das ist kein Fehler, nur der fehlende Text. MMbg kann die Spielstärke von gnubg **nicht abfragen**: Sie wird in gnubg selbst eingestellt (z. B. `set evaluation chequerplay evaluation plies 2`). Wer im Log festhalten will, gegen welche Stufe gespielt wurde, trägt sie hier passend von Hand ein und speichert mit S.

Der gnubg-Zug wird immer gespielt. MMbg sucht die Antwort von gnubg in seiner eigenen Zugliste. Bei manchen Paschen gibt es einige hundert legale Züge, und die Liste kann voll sein (PC3 und PicoCalc 500 Züge). Fehlt der gnubg-Zug deshalb, setzt MMbg ihn oben in die Liste und spielt ihn; das Log vermerkt `--- ext move not in the full list, taken: …`. Nur eine Antwort, die sich nicht lesen oder spielen lässt, überlässt die Entscheidung der Ersatz-Engine (Zeile `!!!` im Log).

> gnubg 1.08.003 vertauscht im `external`-Modus den Spielstand bei Zug- und Doppel-Anfragen. MMbg gleicht das aus. Der Fehler ist an die gnubg-Entwickler gemeldet.

11. Lange Testläufe (Long Runs)

Für Stärkevergleiche spielt der Computer beliebig viele Matches gegen sich selbst oder gegen gnubg.

1. Engines wählen, z. B. `A4E` (big gegen gnubg) oder `A45` (big gegen deep).
2. Modus **C4** (Turbo) starten: ohne Brettanzeige, endlos Matches, eine Statuszeile mit Zwischenständen. **C3** läuft mit Brett, aber ohne Pausen.
3. Ist **Log** an, steht jeder Zug in der Log-Datei im Log-Ordner.
4. Mit **jeder Taste** stoppen.

Hilfen für faire Vergleiche (Optionsseite):

- **U duplicate:** Jedes Match wird mit denselben Würfeln und getauschten Seiten noch einmal gespielt. Das gleicht Würfelglück aus.
- **R dice seed:** ein fester Startwert für reproduzierbare Würfel. 0 nimmt den Startwert aus der Uhr.

Auswertung:

- Am Matchende schreibt das Log `### MATCH` und `### total: games … matches …`.
- Der Titel zeigt den Matchzähler `M a:b`.
- Beispiele: Auf dem PC3 ergab ein Nachtlauf über 13 Stunden 117 Matches big gegen gnubg 2-ply (46,2 %). Auf dem PicoCalc mit Pico 2 W liefen 450 Matches in 8,6 Stunden (Teil 2, P7). Auf dem Colour Maximite 2 liefen 404 Matches big gegen big in 10,4 Stunden (Kapitel 15).

12. Optionen

12.1 Schnellmenü (Taste O)

```
1 anim+ 2 pips+ 3 hint- 4 flip+ 5 swap+ 8 mouse+ 9 mat+ 0 sound-  6 save 7 more
```

Eine Taste schaltet die Option um; `+` heißt an, `-` aus.

Taste	Option
1	Animation der Züge
2	Pips im Titel
3	Tipp (bester Zug, Doppler-Empfehlung im Log)
4	Brett spiegeln (Heimfeld links)
5	Farben tauschen (O rot, X hell)
8	Maus an/aus (PC3, CMM2, PC1)
9	`.mat`-Aufzeichnung (wirkt ab dem nächsten Match)
0	Geräusche: Würfeln, Ziehen, Aufsetzen, Schlagen (nur mit Animation, braucht `OPTION AUDIO`)
6	Einstellungen speichern (`mmbg_pc3.ini`)
7	Optionsseite

12.2 Optionsseite (O → 7)

Die Seite zeigt **alle** Einstellungen mit ihrem Wert. Eine Taste plus ENTER ändert eine Einstellung, ENTER allein führt zurück ins Spiel.

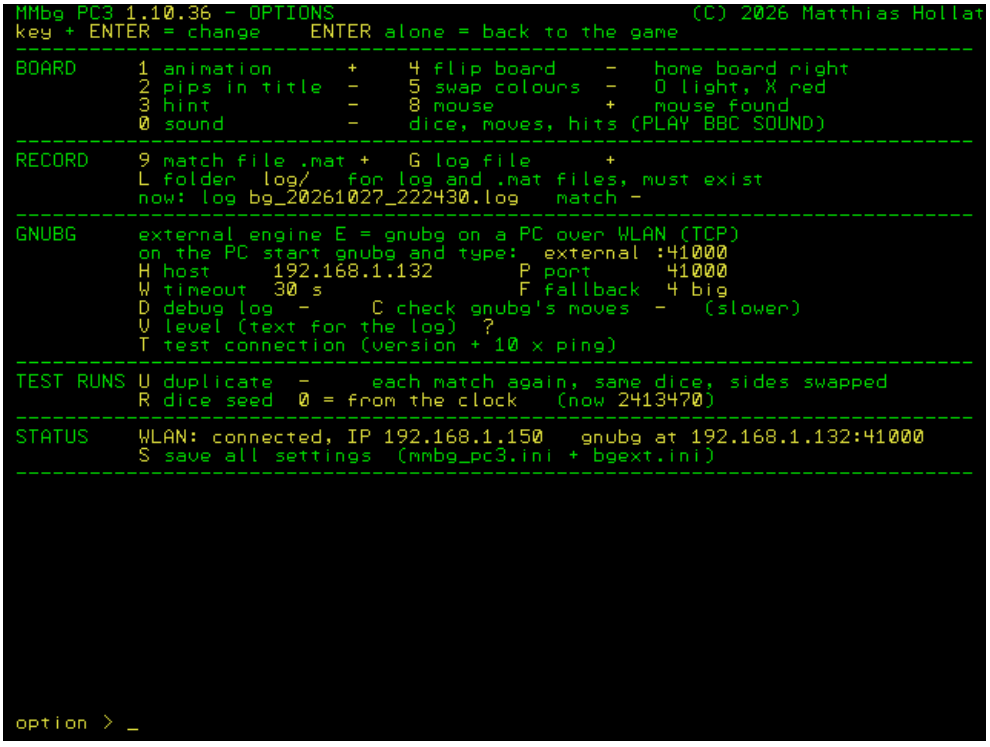


Bild 10: Die Optionsseite (PC3). Logs und `.mat` liegen im Ordner `log/`, gnubg läuft auf dem PC 192.168.1.132.

Bereich	Tasten
BOARD	1–5, 8, 0 wie im Schnellmenü
RECORD	9 `.mat`, G Log an/aus, L Log-Ordner (Kapitel 3.4)
GNUBG	H host, P port, W timeout, F fallback, D debug log, C check, V level, T test
TEST RUNS	U duplicate, R Würfel-Startwert
STATUS	WLAN-Zustand; S speichert alles (`mmbg_pc3.ini` und `bgext.ini`)

13. Befehlsübersicht

Die Taste **H** zeigt alle Befehle und darunter unter ACTIVE, was gerade eingestellt ist:



Bild 11: Die Hilfeseite mit der ACTIVE-Übersicht (HDMI; im Terminal sieht sie gleich aus).

Befehl	Wirkung
`13/7 8/7`, `13 24`, `#3`	ziehen (Kapitel 5.2)
`E` / `U`	Zug beenden / Undo
`R` `D` `T` `P`	würfeln, doppeln, annehmen, aufgeben (an den Doppler-Fragen)
`N` / `M7`	neues Match / Match bis 7
`K`	Doppler an/aus
`SG` / `LG`	Match speichern / laden
GNUbg ID einfügen	Stellung übernehmen
`C0` … `C4`	Computer-Modus
`A45`, `A4E`, `C4E`	Engines je Seite (plus Turbo)
`L`	ganze Zugliste
`D`	Brett-Dump mit GNUbg ID (Kapitel 7.6)
`P`	DEV-Spalte (Terminal)
`O`	Optionen
`G` / `GN`	Log an/aus / neue Log-Datei
`Druck`, `Strg+\`	Bildschirmfoto (Firmware, Kapitel 7.7)

Befehl	Wirkung
`H`	Hilfe mit ACTIVE-Übersicht
`Q`	beenden

14. Dateien

Alle Dateien liegen im Programmordner; der Log-Ordner ist `log/` darunter, sofern nicht anders eingestellt (Kapitel 3.4).

Datei	Ort	Wer schreibt	Inhalt
`mmbg_pc3.ini` (`mmbg_pc1.ini`, `mmbg_pica.ini`)	Programmordner	O → 6 oder S	Oberfläche: anim, pips, hint, flip, swap, mouse, mat, sound, log, match, cube, computer, Engines
`bgext.ini`	Programmordner	Optionsseite S	gnubg/WLAN, Log-Ordner (`logdir=`), Duplicate, Startwert
`mmbg_pc3.sav` (`mmbg_pc1.sav`, `mmbg_pica.sav`)	Programmordner	automatisch, SG	laufendes Match (Zugverlauf, `.mat`-Stand)
`bg_JJJJMMTT_HHMMSS.log`	Log-Ordner	Log an	jeder Zug, Doppler, Ergebnisse
`mmbg_JJJJMMTT_HHMMSS.mat`	Log-Ordner	mat an	Match für gnubg
`SCREEN_JJJJMMTT_HHMMSS.BMP` / `.PNG`	Wurzel der SD-Karte	Druck, Strg+\ (Firmware)	Bildschirmfoto
`bglog_err.txt`	Programmordner	bei Problemen	Schreibfehler des Logs

Inhalt des Pakets

Pfad	Inhalt
`mmbg/`	die drei Programme, Netze, Datenbank, ini-Dateien – diesen Ordner auf die SD-Karte kopieren
`mmbg/log/`	leerer Log-Ordner
`gnubg_start_PC/`	Batch-Dateien zum Start von gnubg auf einem Windows-PC (0-, 1-, 2-ply)
`MMbg_Handbuch.pdf`, `MMbg_Manual.pdf`	Handbuch (Deutsch, Englisch)
`LICENSE.txt`, `README.txt`	Lizenz, Kurzanleitung

15. MMbg auf dem Colour Maximite 2

Seit Version 1.10.34 läuft **dasselbe Programm** `mmbg_pc3.bas` auch auf dem **Colour Maximite 2 (CMM2)**. Möglich macht das die Firmware 7 des CMM2: Sie bringt MMBasic auf den Stand der PicoMite und kann sich mit `OPTION PICOMITE` wie eine PicoMite mit HDMI verhalten. Getestet ist MMbg auf einem CMM2 der Generation 2.

15.1 Voraussetzungen

Was	Anforderung
Rechner	**Colour Maximite 2** (getestet: Generation 2)
Firmware	**MMBasic 7.00.00b3 oder neuer** (erst ab b3 gibt es `OPTION PICOMITE`), getestet mit 7.00.00b4
Kompatibilität	`OPTION PICOMITE HDMI`
Bildschirm	VGA, 640 × 480 (MODE 3 wie auf dem PC3)
Tastatur	USB-Tastatur in der **oberen** USB-Buchse hinten
Maus (optional)	Dual-Mode-Maus (USB/PS/2) in der **unteren** USB-Buchse hinten, mit `OPTION MOUSE 0`
Terminal (optional)	USB-Buchse Typ B (Strom und Konsole), wie beim PC3 80 × 40
WLAN	keines – die Engine E (gnubg) gibt es auf dem CMM2 nicht

Die untere USB-Buchse des CMM2 ist keine echte USB-Buchse: Die Maus läuft dort im **PS/2-Modus**. Deshalb muss es eine Maus sein, die beide Betriebsarten kann, zum Beispiel die Microsoft Basic Optical Mouse.

15.2 Einrichten

Einmalig am Prompt:

```
OPTION PICOMITE HDMI
OPTION MOUSE 0          (nur mit einer Maus in der unteren Buchse)
```

`OPTION LIST` zeigt danach unter anderem:

```
480MHz Colour Maximite 2 G2
MMBasic 7.00.00b4
OPTION PICOMITE PicoMiteHDMIUSB RP2350B, 640x480
```

Die Firmware des CMM2 wird über die obere USB-Buchse und ein USB-A-auf-A-Kabel aufgespielt; das beschreibt das Handbuch des CMM2 (Anhang K).

15.3 Starten

Der Programmordner liegt wie beim PC3 auf der SD-Karte, beim CMM2 also unter `A:/mmbg` (Kapitel 3.2). Gestartet wird so:

```
CHDIR "A:/mmbg"
RUN "mmbg_pc3.bas"
```

Oder bequem über den **Dateimanager FM**: in den Programmordner wechseln und `mmbg_pc3.bas` starten.

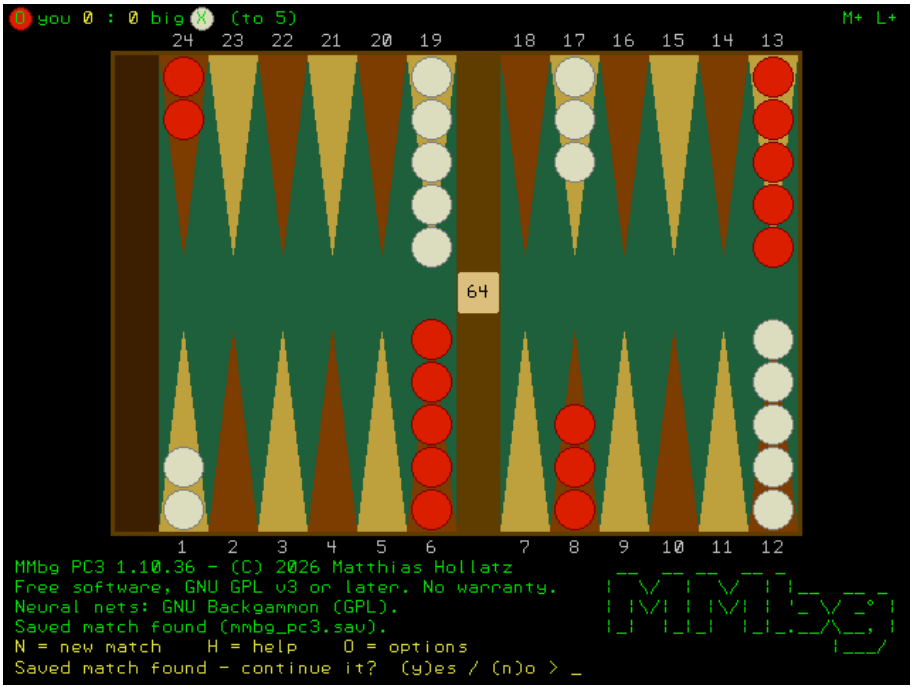


Bild 12: MMbg auf dem Colour Maximite 2 (Screenshot): Start mit gespeichertem Match.

15.4 Was anders ist

- **Enter-Taste:** Der CMM2 liefert bei Enter zwei Zeichen (CR und LF), auch über das Terminal. MMbg stellt beim Start `OPTION CRLF CR` ein und überliest ein übrig gebliebenes LF; ein laufender Computer wird dadurch nicht angehalten.
- **Maus:** MMbg findet die PS/2-Maus von selbst (Mauskanal 2). Ist keine Maus eingerichtet, spielt man wie gewohnt mit der Tastatur.
- **Kein WLAN:** `[WLAN]` erscheint nicht im Titel, die Engine E und der Verbindungstest fehlen. Alles andere – Engines 0 bis 5, Doppler, Undo, Speichern, `.mat`, Log, Testläufe – ist gleich.
- **Tempo:** Der CMM2 (480 MHz) rechnet etwa so schnell wie der PC3 (315 MHz): *big* braucht im Mittel ~0,6 s je Zug, *deep* ~10 s. Ein Turbo-Lauf C4 big gegen big über 10,4 Stunden spielte 404 Matches (~92 s je 5-Punkte-Match) ohne einen Fehler.

16. MMbg PC1: HDMI und VGA ohne PSRAM

`mmbg_pc1.bas` ist MMbg für Boards **ohne PSRAM**, zum Beispiel das **PicoMite HDMI/USB-Board** (hier kurz „PC1“) oder einen **Pico 2 mit VGA-Referenzboard**. Die Oberfläche ist die des PC3: großes Brett mit 640 × 480, Terminal, Maus, Undo, Speichern, `.mat`, Log, Testläufe, alle Engines bis *big* und *deep*. Es fehlt nur das WLAN und damit die Engine E (gnubg).

Die eigentliche Aufgabe war der Speicher: Ohne PSRAM passen die großen Netze (512 KB als Gleitkomma) nicht in den RAM. MMbg PC1 legt sie deshalb mit 16 Bit je Gewicht in den **MODBUFF**, einen festen Bereich im Flash, und liest dort direkt beim Rechnen. Das ist dieselbe Technik wie auf dem PicoCalc (Teil 2, P6 und P7) und spielt genauso stark.

16.1 Voraussetzungen

Was	Anforderung
Rechner	RP2350 mit HDMI oder VGA, ohne PSRAM (getestet: PicoMite HDMI/USB-Board mit RP2350A, Pico 2 mit VGA-Referenzboard)
Firmware	PicoMite **HDMI** bzw. **VGA** V7.00.00 (getestet mit RC6)
MODBUFF	`OPTION MODBUFF ENABLE 192` (siehe 16.2)
Bildschirm	MODE 3, 640 × 480, 16 Farben
Tastatur	USB-Tastatur (HDMI-Board) bzw. die Tastatur des VGA-Boards, dazu das serielle Terminal
Maus (optional)	USB-Maus am HDMI/USB-Board
WLAN	keines – die Engine E und der Verbindungstest fehlen

16.2 Einrichten: MODBUFF 192

Einmalig am Prompt:

```
OPTION MODBUFF ENABLE 192
```

> **Achtung:** Jede Änderung dieser Option **formatiert das Laufwerk A:**. Was dort liegt, vorher sichern. Der Programmordner auf der SD-Karte (B:) bleibt unberührt.

Die Datei `bgnet16.mb` (144 192 Byte) braucht einen MODBUFF von mindestens **152 KB**. Empfohlen sind **192 KB**, denn so viel brauchen auch die meisten MOD-Musikdateien. Mit dem Standardwert 128 KB meldet die Firmware „File too large for modbuffer“, und MMbg spielt nur mit den kleinen Netzen (Engine 3).

Um das Füllen des MODBUFF muss man sich nicht kümmern: MMbg prüft beim Start dessen Inhalt mit den Prüfsummen aus `bgpk16.dat`. Stimmt er nicht, etwa beim ersten Start, lädt MMbg `bgnet16.mb` aus dem Programmordner selbst hinein (`FLASH MODBUFF LOAD`, 1–2 Sekunden). Nur dann wird das Flash beschrieben.

> **Firmware-Wechsel:** Beim ersten Start nach dem Aufspielen setzen die Builds PicoMiteHDMI und PicoMiteRP2350VGA (ab RC6) alle Optionen zurück und löschen Flash und Programmspeicher. Vorher mit `OPTION DISK SAVE` sichern, danach `OPTION MODBUFF ENABLE 192` wieder setzen.

Beispiel für einen Pico 2 mit VGA-Referenzboard (`OPTION LIST`):

```
PicoMiteVGA MMBasic RP2350A Edition V7.00.00RC6
OPTION RESOLUTION 640x480 @ 315000KHz
OPTION SDCARD GP13, GP10, GP11, GP12
OPTION COMPILE ON
OPTION MODBUFF ENABLE  192
```

`OPTION COMPILE ON` funktioniert mit MMbg PC1.

16.3 Starten

```
CHDIR "B:/mmbg"
LOAD "mmbg_pc1.bas"
RUN
```

Im Terminal erscheinen nach den kleinen Netzen die Zeilen `Checking big nets in the MODBUFF ...` und, nur beim ersten Mal, `Loading bgnet16.mb into the MODBUFF (flash) ...`. Danach meldet MMbg `Big nets in the MODBUFF (flash), engine 4.` und den freien RAM.

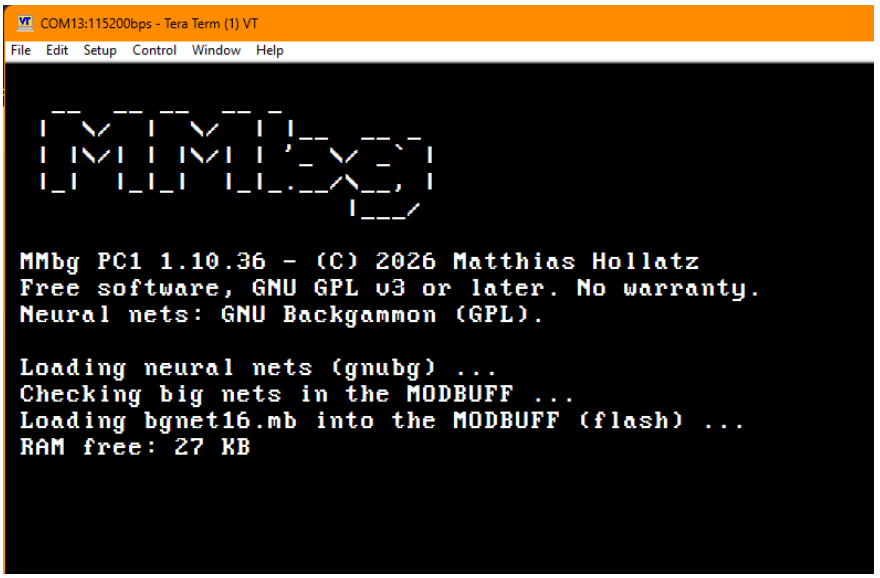


Bild 13: Erster Start von MMbg PC1 im Terminal: Die großen Netze werden im MODBUFF geprüft und einmalig hineingeladen.

Meldet `RUN` „Not enough memory“, hilft `OPTION SYMBOLS OFF` als erste Programmzeile: Das spart 3 bis 5 KB, aber MMBasic findet Variablen dann nur halb so schnell, und der Computer denkt doppelt so lange. Mit Firmware RC6 war das in den Tests nicht nötig.

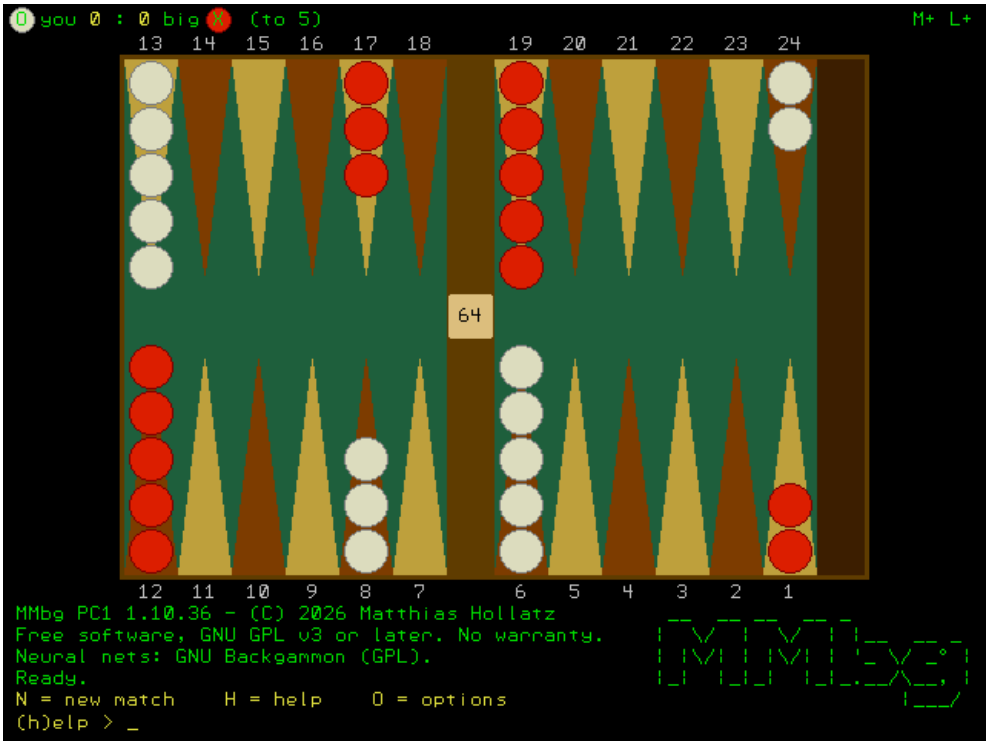


Bild 14: MMbg PC1 auf einem Pico 2 mit VGA-Referenzboard (Screenshot) – dieselbe Oberfläche wie beim PC3.

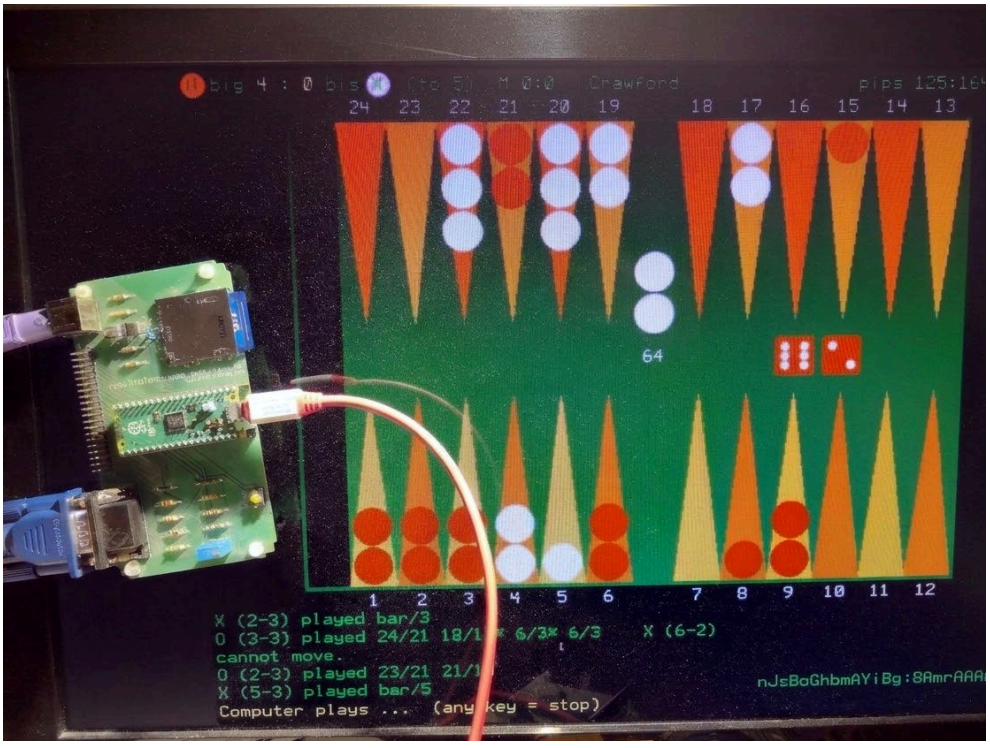


Bild 15: Das VGA-Referenzboard mit Pico 2 im Einsatz (big gegen big, Crawford-Spiel).

16.4 Was anders ist als beim PC3

	PC3	PC1 / VGA
Große Netze	Gleitkomma im PSRAM	16 Bit im MODBUFF (Flash), direkt gelesen
Kleine Netze	Gleitkomma	erste Schicht 16 Bit gepackt
Animation	mit Bildpuffer	**VGA:** Bildpuffer F ohne Platz im RAM, flüssig wie beim PC3. **HDMI:** kein Platz für den Bildpuffer (er bräuchte 150 KB RAM), das Feld unter dem fliegenden Stein wird nachgezeichnet; das sieht gleich aus, der Mauszeiger kann dabei leicht flackern
Zugliste	500 Züge	300 Züge (nur wenige Pasche haben mehr, → 10.2)
Undo	300 Zuganfänge	die letzten 40
`.mat`	400 Aktionen je Spiel	120 (reicht für normale Spiele; in sehr langen Spielen fehlt der Rest in der `.mat`-Datei)
Log-Rang in der Zugliste	alle Züge	die ersten 20
WLAN, Engine E	ja	nein; die Optionsseite hat keinen GNUBG-Bereich
Hilfe	–	zusätzlich eine Zeile `RAM … free now, lowest while thinking …`

Gemessen auf dem HDMI/USB-Board (RP2350A): *big* braucht im Mittel ~0,4 s je Zug. Während des Rechnens bleiben ~16 KB RAM frei.

17. Hinweise und Grenzen

- **Welche Fassung:** `mmbg_pc3.bas` für den PC3 und den Colour Maximite 2 (Kapitel 15), `mmbg_pc1.bas` für HDMI/VGA ohne PSRAM (Kapitel 16), `mmbg_pica.bas` für den PicoCalc (Teil 2).
- **Große Programme:** Ein Programm, das ohne `, C` den Programmspeicher sprengt, kann in den Release Candidates das Flash beschädigen („Program damaged: symbol table“). Die veröffentlichten Programme sind dafür klein genug; mit `, C` geladen bleibt zusätzlich Platz.
- **Programmabbruch mit Strg+C:** Die Farbpalette bleibt verstellt, `MAP RESET` stellt sie zurück. Bei normalem Ende mit `Q` passiert das automatisch.
- **Animation:** MMbg zeichnet den Hintergrund des fliegenden Steins aus einem Bildpuffer (~150 KB PSRAM). Fehlt der Puffer, zeichnet es das Feld Stück für Stück neu; das ist langsamer, sieht aber gleich aus.
- **Geräusche** gibt es nur mit Animation (C0 bis C2). Ohne `OPTION AUDIO` bleibt die Option aus und das Log meldet „no audio output“. Die Klänge (Würfeln, Ziehen, Aufsetzen, Schlagen) sind aus Rauschen gebaut. Wer bessere Einstellungen findet, etwa andere Hüllkurven oder Lautstärken, meldet sie gern im TheBackShed-Forum. Die Werte stehen im Quelltext in `GuSnd` und `GuSndOn`.
- **Nur Matchspiel** bis 15 Punkte, kein Geldspiel. Undo wirkt nur im laufenden Spiel.
- **Eine eingefügte GNUbg ID** beendet die `.mat`-Aufzeichnung für dieses Match, weil eine beliebige Stellung im `.mat`-Format nicht darstellbar ist.

18. Ein Flug durch den Quellcode

Für alle, die in `mmbg_pc3.bas` hineinschauen oder es ändern wollen: ein kurzer Rundgang. Das Programm hat rund 6 300 Zeilen und 263 Routinen. MMbg PC1 und MMbg PiCa entstehen daraus mit Skripten, die genau festgelegte Stellen ersetzen (andere Tabellengrößen, Netze im MODBUFF, LCD-Oberfläche); die Routinen heißen dort gleich. Die Namen verraten den Bereich:

Vorsilbe	Bereich
`Bg…`	Spiel: Regeln, Eingabe, Zuggenerator, Bewertung, Doppler, Match, Log, Speichern
`Nn…` , `Z…` , `Bo…`	kleine Netze, große Netze, Bear-off-Datenbank
`Ex…`	gnubg über WLAN (TCP), Optionsseite
`Gu…`	Grafik, Maus, Optionen, Geräusche
`Gid…` , `BgId…`	GNUbg ID
`BgMat…`	`.mat`-Aufzeichnung

18.1 Start

1. **Globale Variablen** (`DIM …`): Brett `bd(0..25)` (+n = O-Steine, −n = X-Steine, `bd(0)` Bar von O, `bd(25)` Bar von X), Würfel `dw()` , Zuglisten `gm$()` / `gv()` , Netzgewichte, Bildschirmzustand `yb()` .
2. `MODE 3` , `GuPal` (Farben), `BgLogo` (Startbild), `OPTION CRLF CR` (nur CMM2, sonst übersprungen).
3. **Laden:** `NnLoad` (kleine Netze), `BoOpen` (Bear-off-Datenbank), `ExIni` (gnubg-Einstellungen), `BgLogDir` (Log-Ordner anlegen), dann `ZLoad` (große Netze, nur wenn der Speicher am Stück reicht).
4. `GuInit` (Bildpuffer, Farben), `GuCfgLoad` (`mmbg_pc3.ini`), `GuMsInit` (Maus: erst USB, sonst PS/2 auf Kanal 2), Frage nach dem gespeicherten Match (`BgLoadG`).

18.2 Die Hauptschleife

```
DO
  BgShow                Terminal und Bildschirm zeichnen (rechnet die Zugliste)
  Eingabe:  Computer -> bester Zug gm$(1)   |   Mensch -> GuIn$ (Tastatur, Maus)
  Befehl?   H, O, N, M7, C1, A45, SG, LG, U, GNUbg ID ...  -> ausführen
  Zug?      BgInput -> BgTry -> BgMove      (prüfen und ziehen)
            GuAnim -> GuMv -> GuFly        (Animation, Geräusche)
            BgNext -> BgAuto -> BgTurn     (Seitenwechsel, Doppler, Würfeln)
  Spielende? BgEndGame                     (Punkte, Log, .mat, nächstes Spiel)
LOOP
```

18.3 Wie der Computer seinen Zug findet

1. `BgTurn` : erst `BgCubeTurn` (Doppler vor dem Wurf, `BgCube` rechnet die Match-Gewinnchancen wie gnubg 0-ply), dann `BgRoll` .
2. `BgPrep` ruft den **Zuggenerator** `BgGen` : Er probiert alle Würfelreihenfolgen und Startpunkte mit einem eigenen Stapel statt Rekursion (der Stapel reicht nur für wenige Ebenen). `BgMx` prüft vorher, wie viele Würfel höchstens spielbar sind.

3. Jeder fertige Zug landet in `BgLeaf`; doppelte Endstellungen werden verworfen. `BgEval` bewertet die Stellung: Formel (Engines 0–2) oder Netz über `NnEval` → `NnFwd` (klein), `ZFwd` (groß), `NnBo` (Bear-off-Datenbank).
4. `BgSort` sortiert, `ZBest` rechnet die besten 12 mit dem großen Netz nach, `ZThink` schaut bei *deep* einen Wurf voraus, `BgBook` setzt bei Bedarf den Buchzug, `ExMove` fragt gnubg (`ExMvKey$` spielt die Antwort von gnubg probeweise und sucht sie in der Liste oder fügt sie hinzu, wenn die Liste voll ist).
5. Platz 1 der Liste wird gespielt.

18.4 Weitere Bausteine

Routinen	Aufgabe
`BgGid$`, `BgIdGet`, `BgIdPut`	GNUbg ID erzeugen, prüfen, übernehmen
`BgHist`, `BgBack`	Zugverlauf (eine ID je Zuganfang) und Undo
`BgSaveG`, `BgLoadG`	gespeichertes Match schreiben und lesen
`BgMatNew`, `BgMatTurn`, `BgMatGame`	`.mat` im Format von gnubg
`BgLog`, `BgLogTurn`, `BgLogDir`	Log-Datei, eine Zeile je Zug, Log-Ordner
`ExAsk$`, `ExBoard$`, `ExCube`, `ExMove`	Gespräch mit gnubg über TCP (FIBS-Brettzeile)
`GuFull`, `GuDraw`, `GuPt`, `GuDice`	Bildschirm neu oder nur Geändertes zeichnen
`GuFly`, `GuNear`, `GuRep`	fliegender Stein, Mauszeiger, Rückfall ohne Bildpuffer
`GuHit$`, `GuShort$`	Mausklick und Kurzeingabe in einen Zug übersetzen
`GuSnd`, `GuSndOn`	Geräusche mit `PLAY BBC SOUND`
`BgHelp`, `ExMenu`, `GuOpt`	Hilfe, Optionsseite, Schnellmenü

Teil 2 – MMbg PiCa (PicoCalc)

> **Neu in 1.10.36:** Die großen Netze liegen jetzt im **MODBUFF** (Flash) wie bei MMbg PC1, nicht mehr in einem Zeilenpuffer im RAM mit Nachladen von der SD-Karte. Dadurch sind Zugliste, Undo und `.mat` so groß wie beim PC3. Dazu kommen Bildschirmfotos über die Firmware und kürzere Eingabezeilen.

Teil 2 beschreibt nur, **was auf dem PicoCalc anders ist**. Für das Spiel selbst – Ziehen, Doppler, Undo, Speichern, `.mat`, Log, GNUbg ID, Engines, gnubg über WLAN und Testläufe – gilt Teil 1 unverändert; die Verweise (z. B. „→ 5.2“) zeigen auf die Kapitel dort.

P1. MMbg PiCa auf einen Blick



Bild 16: MMbg PiCa auf dem PicoCalc (Pico 2 W), Computer gegen Computer.

- **Ein Programm** (`mmbg_pica.bas`) für den PicoCalc mit **Pico 2** (Firmware PicoMite RP2350) oder **Pico 2 W** (Firmware WebMite RP2350). Es erkennt die Firmware beim Start selbst.
- **Dasselbe Spiel wie auf dem PC3:**
  - alle Engines bis *big* und *deep* (→ 8.1),
  - Doppler, Crawford, Speichern und Fortsetzen, mehrstufiges Undo, GNUbg IDs, `.mat`-Aufzeichnung, Log, Geräusche, Bildschirmfotos (→ 5–7),
  - Computer-Modi C2 bis C4 und lange Testläufe (→ 8.3, 11),
  - gnubg als Gegner über WLAN und das Terminal über Telnet (nur Pico 2 W, → 10).
- **Dieselbe Spielstärke wie der PC3:** Die Netze sind dieselben gnubg-Gewichte, nur mit 16 Bit gespeichert; das verliert nichts Messbares (P7). Seit 1.10.36 ist auch die Zugliste wie beim PC3 500 Züge lang, die PiCa wählt also aus genau denselben Zügen (P6.3).
- Braucht **kein PSRAM**: Die großen Netze liegen im MODBUFF (P6). Nötig ist einmalig `OPTION MODBUFF ENABLE 192`.

P2. Voraussetzungen und Einrichten der Firmware

Was	Anforderung
Rechner	PicoCalc mit **Pico 2** (RP2350, ohne PSRAM) oder **Pico 2 W**
Firmware	**PicoMite RP2350** oder **WebMite RP2350**, **V7.00.00** (getestet mit RC6 auf dem Pico 2 W)
MODBUFF	`OPTION MODBUFF ENABLE 192` (P6)
SD-Karte	Laufwerk B: für den Programmordner (Programm, Netze, Logs)
Terminal (optional)	VT100, 80 × 40, über USB-Seriell oder Telnet (WebMite)
WLAN (optional)	nur Pico 2 W: gnubg als Gegner (→ 10), Terminal über Telnet

Die Firmware einmalig am Prompt einrichten:

```
OPTION RESET PICOCALC               (LCD, Tastatur, SD-Karte, Ton)
OPTION COMPILE ON                   (wichtig fürs Tempo)
OPTION MODBUFF ENABLE 192           (formatiert A: - vorher sichern!)
OPTION PRINTSCREEN FORMAT PNG       (wenn Bildschirmfotos als PNG gewünscht)
OPTION WIFI "MeinNetz", "Passwort"  (nur Pico 2 W)
OPTION TELNET CONSOLE ON            (nur Pico 2 W, wenn gewünscht)
```

P3. Installation und Start

P3.1 Ein Ordner für alles

Wie beim PC3 (→ 3.1) liegen alle Dateien in **einem Ordner**, zum Beispiel `B:/mmbg`; Name, Laufwerk und Tiefe sind frei. Das Paket enthält die Dateien für alle Geräte. Jede Fassung nimmt sich nur, was sie braucht (P8); derselbe Ordner funktioniert also überall.

Ordner	Inhalt
`mmbg` (beliebiger Name)	Programme, Netze, Datenbank, ini-Dateien, gespeichertes Match
`mmbg/log`	Logs und `.mat`-Dateien; MMbg legt den Ordner beim Start an (→ 3.4)

P3.2 Starten

```
CHDIR "B:/mmbg"
LOAD "mmbg_pica.bas", C
RUN
```

Auf dem PicoCalc ist `, C` **nötig**: Ohne die Kommentare bleibt genug Programmspeicher.

Das Startbild zeigt nacheinander:

Zeile	Bedeutung
`Nets: bgnet345.dat (… ms).`	kleine Netze geladen

Zeile	Bedeutung
`Checking big nets in the MODBUFF ...`	MMbg prüft die großen Netze im MODBUFF (Terminal)
`Loading bgnet16.mb into the MODBUFF ...`	nur beim ersten Start oder nach einem Firmware-Wechsel: `bgnet16.mb` wird ins Flash geladen (1–2 s)
`Big net: MODBUFF (flash).`	große Netze bereit, Engine 4
`Big net: MODBUFF 192 needed - engine 3.`	kein MODBUFF eingerichtet: nur die kleinen Netze (P2)
`Bear-off db: bgbo15.dat (with gammons).`	Endspiel-Datenbank
`Settings: mmbg_pica.ini.`	Einstellungen geladen
`RAM free: … KB.`	freier Speicher nach dem Start

Danach wartet MMbg auf **N** (neues Match). Ist ein Match gespeichert, fragt es, ob es fortgesetzt werden soll (→ 6.2).



Bild 17: Links der Start im Terminal (die großen Netze werden einmalig in den MODBUFF geladen), rechts das LCD mit der kurzen Frage vor dem Wurf.

P4. Bildschirm, Terminal und Bedienung

P4.1 LCD (320 × 320)

- **Titelzeile** (40 Zeichen): Spieler, Stand, Matchlänge und die Schalter wie beim PC3 (→ 4.1): `W` WLAN, `M±` `.mat`, `L±` Log.
- **Brett:** Punkt 23 px, Stein 19 px, Bar und Ablage. Wie beim PC3 liegen die Würfel rechts vom Spieler aus gesehen.
- **Log:** die letzten 4 Meldungen zu 40 Zeichen.
- **Prompt:** ganz unten, 40 Zeichen. Alle Eingabefragen sind höchstens 39 Zeichen lang und deshalb kürzer als beim PC3, z. B. `O: (r)oll / (d)ouble to 2 >`, `X: O doubles to 2 (t)ake/(p)ass >` oder `O wins match 5-2! ENTER=new >`. Die Bedeutung der Nummern steht auf der Hilfeseite.

P4.2 Terminal

Über USB-Seriell oder Telnet zeigt das Terminal **dasselbe ASCII-Brett wie beim PC3** (→ 4.2), 80 × 40, mit Zugliste, DEV-Spalte und GNUbg ID. MMbg stellt das Terminal beim Start auf 80 × 40, wenn das Terminalprogramm das zulässt (Tera Term, PuTTY). Eingaben gehen über die PicoCalc-Tastatur und das Terminal gleichermaßen.

P4.3 Textseiten (Hilfe, Optionen, Liste, Dump)

Alle Textseiten haben **40 Spalten in Font 1 (8 × 12)**, 26 Zeilen. Längere Seiten sind auf **zwei Seiten** verteilt:

Seite	Inhalt	Blättern
H Hilfe 1/2	alle Befehle (→ 13)	Pfeil runter oder beliebige Taste → Seite 2, Q zurück
H Stand 2/2	der aktuelle Stand: Match, Doppler, Spieler, Computer-Modus, Optionen, Log-Ordner und -Datei, `.mat`-Datei, gespeichertes Match, Netze, Bear-off-Datenbank, WLAN, gnubg, GNUbg ID	Pfeil hoch → Seite 1, jede andere Taste zurück
O → 7 Optionen 1/2	Brett und Aufzeichnung: 1–5, 0, 9, G und L mit dem Log-Ordner in einer eigenen Zeile	Pfeil runter → Seite 2
O → 7 Optionen 2/2	gnubg und Testläufe: H, P, W, F, D, C, V, T, U, R und der WLAN-Zustand	Pfeil hoch → Seite 1
L Zugliste	20 Züge je Seite	Taste → weiter, Q zurück
D Dump	Brett zweispaltig, Würfel, **GNUbg ID**, Ort des großen Netzes (MODBUFF)	Taste → zurück

Auf der Optionsseite gelten **alle Tasten auf beiden Seiten**; nach einer Änderung erscheint die Seite mit dem geänderten Wert. **S** speichert alles (`mmbg_pica.ini` und `bgext.ini`). Das Terminal zeigt dieselben Seiten, die Eingabezeile direkt darunter.

GNUbg ID zum Abschreiben: Taste **D** und die Stand-Seite 2/2 zeigen die ID auf dem LCD. Im Terminal steht sie wie beim PC3 rechts unten. Eingefügt wird eine ID wie gewohnt am Prompt (→ 7.4).

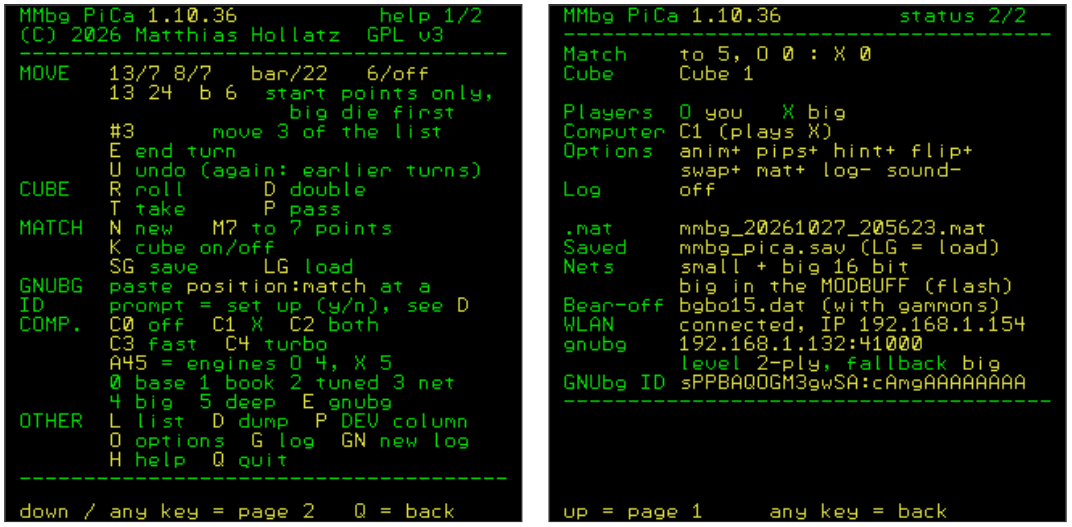


Bild 18: Textseiten auf dem LCD: Hilfe 1/2 und Stand 2/2 (Netze „big in the MODBUFF (flash)“).

P4.4 Schnellmenü (Taste O)

Wie beim PC3 (→ 12.1), angezeigt im Log-Bereich des LCD und ohne Maus-Schalter: `1`–`5`, `9`, `0`, `6 save`, `7 more`.

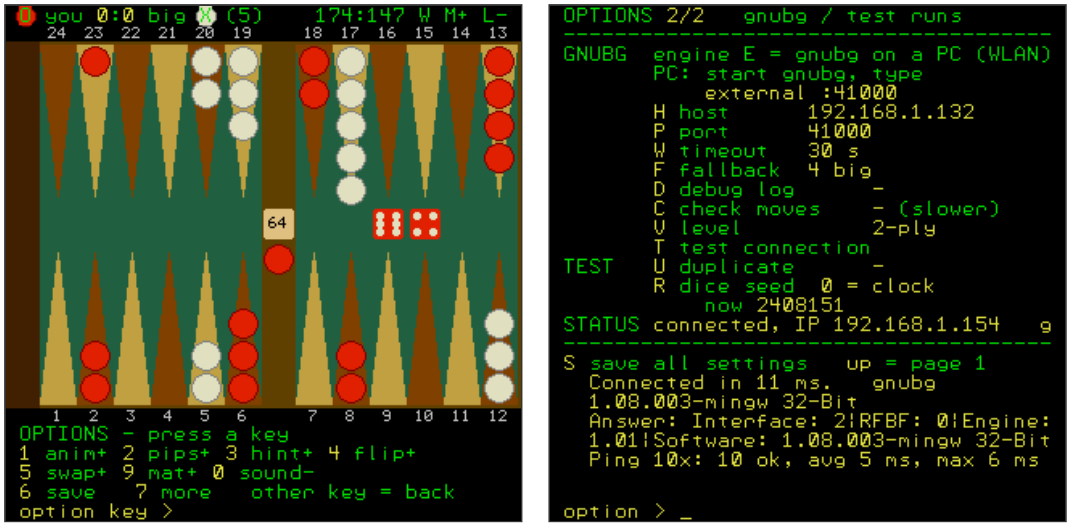


Bild 19: Links das Schnellmenü (Taste O) im Log-Bereich, rechts die Optionsseite 2/2 nach einem geglückten Verbindungstest (Pico 2 W).

P4.5 Turbo-Modus C4 auf dem LCD

Im Turbo-Modus (→ 8.3, 11) schaltet MMbg das LCD **schwarz** und das Hintergrundlicht auf die **kleinste Stufe**, damit über Stunden nichts unverändert auf dem Bildschirm steht. Jede Taste stoppt den Lauf; das Hintergrundlicht kehrt auf seinen vorigen Wert zurück und das Brett wird neu gezeichnet.

Das Hintergrundlicht stellt der Tastatur-Controller des PicoCalc ein, der Werte von 16 bis 240 annimmt; MMbg benutzt `OPTION BACKLIGHT LCD`, das den Wert auch im Flash speichert. Wird ein Turbo-Lauf mit **Strg+C** beendet, entfällt das Zurückstellen: das Licht dann mit `OPTION BACKLIGHT LCD n` (n von 16 bis 240) wieder einstellen.

P5. Unterschiede zum PC3

	PC3	PiCa
Bildschirm	HDMI 640 × 480	LCD 320 × 320
Textseiten	80 Spalten	40 Spalten, Hilfe und Optionen je 2 Seiten
Eingabezeilen	ausführlich	kurz, höchstens 39 Zeichen (P4.1)
Maus	ja	nein
Bildschirmfoto	Druck, Strg+\ (→ 7.7)	ebenso, das Bild hat 320 × 320 Punkte
Animation	mit Bildpuffer (PSRAM)	ohne Bildpuffer, das Feld wird nachgezeichnet; sieht gleich aus
GNUbg ID	rechts unten im Log	im Terminal, bei D und auf der Stand-Seite 2/2
Zugliste, Undo, `.mat`, Log-Rang	500 Züge, 300 Zuganfänge, 400 Aktionen, alle Züge	**gleich**
Große Netze	Gleitkomma (8 Byte) im PSRAM	**16 Bit im MODBUFF** (Flash), direkt gelesen (P6)
Kleine Netze	Gleitkomma	erste Schicht 16 Bit gepackt
Turbo C4	ohne Brettanzeige	LCD dunkel, Hintergrundlicht kleinste Stufe (P4.5)
WLAN / Telnet	ja	nur Pico 2 W

Alles andere – Spiel, Engines, Doppler, Speichern, `.mat`, Log und gnubg – ist **gleich**.

P6. Netze und Speicher auf dem PicoCalc

P6.1 Die Aufgabe

Das große gnubg-Netz hat 250 Eingaben × 128 Neuronen = **32 000 Gewichte**, und es gibt zwei davon (Kontakt und crashed, → 9.2). Als Gleitkommazahlen (8 Byte je Zahl, wie beim PC3) bräuchten beide zusammen 512 KB. Der PicoCalc hat kein PSRAM, und im RAM bleiben nach dem Laden von MMbg nur einige hundert KB.

P6.2 Gepackte Gewichte

Jedes Gewicht wird als vorzeichenlose 16-Bit-Zahl `u` gespeichert und beim Rechnen zurückgewandelt:

```
Gewicht = lo + u × q
```

Die Skala gilt **je Eingangszeile**: `lo` ist das kleinste Gewicht der Zeile, `q` = (größtes − kleinstes) / 65 535. Der Rundungsfehler je Gewicht beträgt höchstens q/2, typisch 0,0003 bis 0,0006 (P7). Ein Netz braucht so 64 000 Byte, beide zusammen mit Skalen und Escape-Tabellen **144 192 Byte**. Diese Daten stehen in `bgnet16.mb`, die Prüfsummen in `bgpk16.dat`.

P6.3 Die Netze im MODBUFF

Der **MODBUFF** ist ein fester Bereich im Flash, den die Firmware mit `OPTION MODBUFF ENABLE 192` reserviert. MMbg legt dort `bgnet16.mb` ab und liest die Gewichte beim Rechnen **direkt aus dem Flash**. Im RAM liegt davon nichts.

- **Beim Start** prüft MMbg den Inhalt mit den Prüfsummen. Stimmt er nicht (erster Start, Firmware neu aufgespielt), lädt es `bgnet16.mb` mit `FLASH MODBUFF LOAD` selbst hinein (1–2 s). Nur dann wird das Flash beschrieben.
- **Beim Rechnen** liegen die Gewichte zeilenweise nach Eingängen: Eine Zeile enthält die 128 Gewichte eines Eingangs zu allen Neuronen (256 Byte). In einer echten Stellung sind nur etwa 50 der 250 Eingänge ungleich null, also braucht eine Bewertung nur diese ~50 Zeilen. Jede Zeile wird mit drei schnellen Firmware-Befehlen verarbeitet:

```
MEMORY UNPACK (Zeile aus dem Flash auspacken) -> MATH SCALE (× Eingang × q) -> MATH
C_ADD (aufsummieren)
```

- **Ohne MODBUFF** (oder mit nur 128 KB) meldet das Startbild `Big net: MODBUFF 192 needed - engine 3.`; dann spielen die Engines mit den kleinen Netzen.

Die Rechenzeit liegt wie beim PC3 bei ~0,5 s je Zug. Weil die Netze keinen RAM mehr belegen, sind Zugliste, Undo und `.mat` so groß wie beim PC3.

Gleich stark wie der PC3: Bisher war die Zugliste der PiCa auf 300 Züge begrenzt. Bei manchen Paschen gibt es mehr legale Züge, dann fehlte der Rest in der Auswahl. Mit 500 Zügen bewertet die PiCa jetzt in jeder Stellung genau dieselben Kandidaten wie der PC3, mit demselben Vorfilter und denselben Netzen. Der einzige Unterschied bleibt die 16-Bit-Rundung der Gewichte, und die liegt weit unter allem, was sich messen lässt (P7.3).

> Bis Version 1.10.35 hielt die PiCa die Netze in einem **Zeilenpuffer im RAM** und lud fehlende Zeilen von der SD-Karte nach. Das ging, ließ aber wenig RAM übrig. Auf die MODBUFF-Lösung kam es bei der Entwicklung von MMbg PC1 (Kapitel 16); beide Fassungen nutzen jetzt denselben Netz-Code.

P6.4 Kleine Netze

Die erste Schicht der kleinen Netze (Kontakt, crashed, Rennen: 200 × 16 bzw. 200 × 8 Gewichte) ist ebenfalls mit 16 Bit gepackt, mit einer Skala je Netz. Sie liegt im RAM. Die zweite Schicht ist klein und bleibt Gleitkomma.

P7. Die 16-Bit-Netze: Rundung und Spielstärke

P7.1 Woher die Rundung kommt

Mit 16 Bit kann ein Gewicht 65 536 verschiedene Werte zwischen dem kleinsten und dem größten Gewicht **seiner Zeile** annehmen. Weil jede Zeile ihre eigene Skala hat, sind die Stufen fein: Eine typische Zeile überspannt etwa 45 (Kontaktnetz) bis 80 (crashed-Netz) Einheiten, eine Stufe ist also etwa 0,0007 bis 0,0012 groß, und der Rundungsfehler eines Gewichts beträgt höchstens eine halbe Stufe – typisch 0,0003 bis 0,0006, in der breitesten Zeile 0,0036. In einer Bewertung

werden etwa 50 Zeilen × 128 Neuronen zusammengezählt; die Fehler haben zufällige Vorzeichen und heben sich weitgehend auf, und die S-Kurven der verdeckten Schicht dämpfen sie zusätzlich.

P7.2 Wie stark ändert sich ein Stellungswert?

Die 16-Bit-Bewertung wurde mit einem Python-Nachbau der MMbg-Bewertung (Eingaben, beide Netzgrößen, Zuggenerator, Vorauswahl) und den Logs echter Läufe auf dem PicoCalc gegen die Bewertung des PC3 in voller Genauigkeit geprüft:

Prüfung	Ergebnis
1 014 Stellungen eines Laufs gegen gnubg 2-ply	Abweichung der Equity höchstens **0,0011**
2 274 Stellungen des Nachtlaufs	Abweichung höchstens **0,0012**
680 Zugentscheidungen aus 34 Spielen	**679 gleiche Züge**; einmal waren zwei Züge praktisch gleich gut (Unterschied 0,00002) und tauschten die Plätze

Eine Abweichung von 0,001 Equity ändert den Zug nur, wenn zwei Züge fast genau gleich gut sind – und dann ist es kaum von Belang, welcher der beiden gespielt wird.

P7.3 Geschätzter Verlust an Spielstärke

Aus der Prüfung oben: eine vertauschte Entscheidung unter 680, die 0,00002 Equity kostet, ergibt einen mittleren Verlust von etwa **0,00000003 Equity je Zug** (3×10^{-8}). Zum Vergleich: Das große Netz selbst macht gegenüber gnubg 2-ply im Mittel einen Fehler von **0,0026 Equity je Zug** (2,6 mEq, → 8.1). Der Rundungsverlust ist also etwa **ein Hunderttausendstel** des eigenen Fehlers des Netzes – weit unter allem, was sich in Partien messen lässt.

P7.4 Im Spiel gemessen

Ein Nachtlauf auf einem Pico 2 W (16 Bit, big gegen gnubg 2-ply, Matches bis 5, Turbo C4):

	Pico 2 W, 16 Bit	PC3, Gleitkomma
gewonnene Matches	**199 von 450 = 44,2 % ± 4,6 %**	577 von 1236 = 46,7 %
gewonnene Spiele	778 von 1631 = 47,7 %	–
Laufzeit	8,6 h, ~69 s je Match	–
Rechenzeit je Zug	~0,5 s	~0,5 s

Gemessen wurde noch mit dem Zeilenpuffer (bis 1.10.35); die Gewichte und die Rechnung sind dieselben wie heute im MODBUFF. Der Unterschied von 2,5 Prozentpunkten liegt gut innerhalb der statistischen Streuung (etwa eine Standardabweichung); bei 450 Matches ist ein Unterschied dieser Größe schon durch Zufall zu erwarten. Zusammen mit P7.3: **Der PicoCalc spielt so stark wie der PC3.**

P8. Dateien

Datei	PC3 / CMM2	PC1 / VGA	PicoCalc	Inhalt
`mmbg_pc3.bas`	✓			Programm PC3 und CMM2
`mmbg_pc1.bas`		✓		Programm PC1 (HDMI/VGA ohne PSRAM)
`mmbg_pica.bas`			✓	Programm PiCa
`bgnet345.dat`	✓	✓	✓	kleine Netze (Text)
`bgnet02.dat`	✓	✓	✓	Ausgabeschicht der großen Netze
`bgnet0.bin`, `bgnet2.bin`, `bgesc.bin`	✓			große Netze, Gleitkomma
`bgnet16.mb`, `bgpk16.dat`		✓	✓	große Netze 16 Bit und Escape-Tabellen für den MODBUFF, Prüfsummen
`bgbo15.dat`	✓	✓	✓	Bear-off-Datenbank
`bgnet35.dat`, `bgbo10.dat`	✓	✓	✓	Ersatzdateien (→ 3.1)

Die früheren Netzdateien der PiCa (`bgnet0.pk`, `bgnet2.pk`, `bgpk.dat`, `*.p16`, `*.s16`, `bgesc.pk`) werden nicht mehr gebraucht und können gelöscht werden.

Von MMbg selbst angelegt:

- im Programmordner: `mmbg_pica.ini` (Oberfläche, mit `log=`), `bgext.ini` (gnubg, `logdir=`, Testläufe; gemeinsam mit dem PC3), `mmbg_pica.sav` (gespeichertes Match);
- in `log/`: `bg_JJJJMMTT_HHMMSS.log` und `mmbg_JJJJMMTT_HHMMSS.mat`.

P9. Hinweise und Grenzen

- **Nachbild auf dem LCD:** Das Panel des PicoCalc kann ein schwaches Nachbild eines Bildes behalten, das lange unverändert stand, etwa Text am Prompt. Man sieht es später als Schatten auf dunklen Farben, zum Beispiel auf dem Brettfilz. Es verblasst wieder: ausschalten oder eine Weile eine helle Fläche zeigen (`BOX 0,0,320,320,0,RGB(WHITE),RGB(WHITE)`). Deshalb verdunkelt der Turbo-Modus C4 den Bildschirm (P4.5).
- **Würfel:** Im normalen Spiel kommen sie aus `RND`; auf dem RP2350 mischt die Firmware den Zufall laufend aus dem Hardware-Zufallsgenerator neu. Nur bei **Duplicate** (Option U, → 11) sind die Würfel absichtlich aus einem Startwert wiederholbar. Ohne eigenen Wert (Option R) stammt dieser aus der Uhr, und die steht beim Pico 2 nach jedem Einschalten auf 01.01.2024. Für echte Testläufe also einen Startwert setzen.
- **Terminalgröße:** Die automatische Umstellung auf 80 × 40 klappt nur, wenn das Terminalprogramm sie zulässt. Sonst bitte von Hand einstellen.
- **Telnet:** Laut Release-Notes der Firmware behebt RC4 Fehler bei der Telnet-Ausgabe. Fehlen Zeichen im Telnet-Terminal, liegt es vermutlich daran.
- **Strg+C im Turbo-Modus:** siehe P4.5 (Hintergrundlicht).
- **Fehlermeldungen mit Zeilennummer** beziehen sich beim Laden mit `, C` auf das Programm ohne Kommentarzeilen.

Lizenz und Dank

MMbg ist **freie Software** unter der **GNU General Public License v3 oder neuer**: Es darf genutzt, weitergegeben und verändert werden, auch kommerziell. Wer es weitergibt, muss es wieder unter der GPL tun, mit Quelltext. Es gibt keine Garantie.

```
MMbg - Backgammon for MMBasic
Copyright (C) 2026 Matthias Hollatz - Soli Deo gloria
Neural network weights derived from GNU Backgammon,
(C) the GNU Backgammon authors (GPL v3 or later).
```

Dank an

- **Geoff Graham**, den Erfinder von MMBasic und des Maximite,
- **Peter Mather** für die PicoMite- und WebMite-Firmware, die Firmware für PC3, PicoCalc und Colour Maximite 2 und die BBC-Sound-Engine,
- die Autoren von **GNU Backgammon** für Netze, Methoden und die Prüfung über `external`,
- die Community im **TheBackShed-Forum**.

Viel Spaß beim Spielen!