# MMbg Win 1.10.36 – Backgammon on MMBasic for Windows

Addendum to MMbg 1.10.36 · Matthias Hollatz · November 2026

MMbg Win is MMbg for **MMBasic for Windows** (MMB4W) by Peter Mather. Game, engines, net files and controls are the same as in MMbg PC3 1.10.36. This document only describes what is different on Windows: setup, differences to the PC3 version, peculiarities of MMB4W and how MMbg deals with them.

## 1 At a glance

| | **MMbg Win** |
|---|---|
| File | `mmbg_win.bas` (195 KB, with short English comments) |
| Runtime | MMBasic for Windows **V5.07.03b25** ( `MMBasic.exe` , no installation) |
| Screen | graphics window 640 × 480 ( `MODE 8` ), font 8 × 12 |
| Terminal | the CMD window `MMBasic.exe` was started from (display only) |
| Input | keyboard and mouse in the graphics window, **Ctrl+V** pastes from the clipboard |
| Engines | 0 to 5 as on the PC3, **E = gnubg** via TCP (same PC or another one on the network) |
| Sound | option 0, default off; rebuilt with `PLAY SOUND` |
| Speed | big vs big: **13.7 s per 5-point match** (PC3/CMM2 about 90 s) |

## 2 Setup and start

1. **Get MMB4W:** the EXE is only available on TheBackShed forum, thread "MMBasic for Windows - betas", last page (V5.07.03b25 of 2 October 2025). The Fruit of the Shed wiki has an overview and the manual. `MMBasic.exe` needs no installation. If it does not start or DLLs are missing, the wiki recommends the 32-bit Visual C++ runtime.
2. **Take the MMbg package:** unpack the latest published package MMbg 1.10.36 RC. The folder `mmbg` already contains all net files ( `bgnet345.dat` , `bgnet0.bin` , `bgnet2.bin` , `bgnet02.dat` , `bgesc.bin` , `bgbo15.dat` ). Copy `mmbg_win.bas` into this folder and put the folder under **Documents**.
3. **Start from a CMD window:** open `cmd.exe` , make it at least **80 columns × 40 rows**, change to the Documents folder and run `MMBasic.exe` there (with its path if it lives elsewhere).
4. **In the graphics window of MMBasic:**

   ```
   CHDIR "mmbg"
   LOAD "mmbg_win.bas"
   RUN
   ```

   After the load messages MMbg waits 3 seconds (any key skips).

5. **Place the windows side by side:** Windows often puts the graphics window on top of the CMD window. MMbg cannot move windows.

The first start creates `mmbg_win.ini`, `bgext.ini` and, in the sub-folder `log/`, one log per session and the `.mat` files for gnubg. Missing net files only disable the engines that need them.

## 3 Controls – what is different

- **Keys and mouse only work in the graphics window.** The CMD window shows the terminal view (ASCII board, move list, messages) but takes no input. Text can be marked and copied there.
- **Ctrl+V** pastes the first line of the Windows clipboard into the input line. This brings a GNUbg ID from gnubg (or from the CMD window) straight in. After Ctrl+V it takes almost a second ("reading the clipboard ..."). The ID is also found with text before or after it. If it is incomplete, MMbg reports "GNUbg ID incomplete: 26 of 27 characters".
- **Screenshot:** no key of its own; use Windows (e.g. Win+Shift+S).
- **Sound** (options → `0`): dice, slide, landing, hit, tray. Default off. Silent in C3 and C4.
- **Log:** one file per session (the PC3 starts a new one after 1 MB). `GN` still starts a new log by hand.
- `Q` closes the log and the gnubg connection and ends MMbg.

## 4 Playing against gnubg (engine E)

1. Start gnubg on this or another PC (gnubg-cli is enough) and type `external :41000`. gnubg then waits for **one** connection; type it again after each run.
2. In MMbg: options `O`, then `7`. Under `H` enter the address (e.g. `192.168.1.132`, or `127.0.0.1` for the same PC), `P` port 41000, test the connection with `T`.
3. `A4E` (O = big, X = gnubg), then `C2` or `C4`. The playing strength is set in gnubg; `V` is only the label for the log (e.g. "0-ply").
4. For test runs switch on duplicate mode (`U`): every match is replayed with the same dice and the sides swapped.

**Important:** the "timeout" setting has no effect on Windows and is marked "(not used)". If gnubg stops answering without closing the connection, MMbg waits (see section 6.9). If gnubg is closed or the connection drops, the fallback engine takes over.

**Note for analysis:** in duplicate mode the line `### total ... matches O - X` counts by **side**, not by engine. The score of an engine has to be counted per match from the lines `=== duplicate pair ... engines O=... X=...` and `### MATCH to ...`.

# 5 Changes against MMbg PC3 1.10.36

| Area | PC3 | Win |
|---|---|---|
| Screen | `MODE 3`, 16 colours via `MAP` | `MODE 8` + `FONT 1`, true colours via `RGB()` |
| Animation | framebuffer F + `BLIT` | no framebuffer, the background is repaired (`GuRep`) |
| Mouse | `DEVICE(MOUSE ...)`, firmware pointer (`GUI CURSOR`) | `MOUSE(X/Y/L/R)`, Windows pointer |
| Terminal | serial console | CMD window (`OPTION CONSOLE SERIAL`), cursor via own VT100 sequence (`WAt$`) |
| Writing files | direct | console switched to `SCREEN` for a moment (`WFile`) |
| ENTER | code 13 | arrives as code 10 |
| Net computation | `MATH C_ADD` | loop `WAdd` |
| Log | `FLUSH`, free-space check, new log after 1 MB | close/reopen instead of `FLUSH`, no space check, one log per session |
| gnubg | `WEB OPEN TCP STREAM` over WLAN, with timeout | `TCP CLIENT / SEND / RECEIVE`, without timeout |
| Sound | `PLAY BBC SOUND` with envelopes | `PLAY SOUND` noise, envelope via `SETTICK` |
| Waiting for keys | no pause | `PAUSE 10` per round |
| Paste | terminal program | Ctrl+V via PowerShell |
| Screenshot | `OPTION PRINTSCREEN` | Windows |
| Program end | `SUB MM.END` | also cleans up before every `END` |

Everything else – rules, move generator, engines 0–5, bear-off database, cube, match equity table, GNUbg ID, save and load, `.mat` recording – is line for line the code of MMbg PC3 1.10.36.

# 6 Peculiarities of MMB4W and how MMbg deals with them

All points were measured on MMB4W V5.07.03b25 unless stated otherwise. What was only read in the published MMB4W source (version b20) is marked as such.

## 6.1 Colours only via `RGB()`

Raw colour values `&Hrrggbb` appear with red and blue swapped (the checkers were blue instead of red). With `RGB(r, g, b)` everything is right. There is no `MAP`; MMbg keeps the 16 colours as true colours.

## 6.2 Font grid

`MODE 8` gives 640 × 480 but does not clear the window. Depending on the previous state the text used a 16 × 24 grid instead of 8 × 12, and the text pages no longer fitted the window. MMbg therefore sets `FONT 1` and `CLS` explicitly after `MODE 8`.

### 6.3 ENTER arrives as LF

Code 13 did not arrive; MMbg takes code 10 as ENTER (a 10 right after a 13 is ignored).

### 6.4 Terminal in the CMD window – and a trap when writing

`OPTION CONSOLE SERIAL` sends the output to the CMD window MMB4W was started from. VT100 colours and cursor commands work there. But: **in SERIAL mode `PRINT #n` also ends up in the CMD window instead of the file** – the test file had 0 bytes. MMbg therefore switches to `OPTION CONSOLE SCREEN` for a moment before every file write and back afterwards (13 places, 56 `PRINT #`).

### 6.5 `@(x, y)` only in `PRINT`

`a$ = @(x, y)` gives "Invalid function". For the terminal MMbg builds the cursor position itself as the VT100 sequence `ESC[row;columnH` (`WAt$`). At the right edge of the CMD window the status line overwrote the last character of the GNUbg ID; the ID is therefore two columns further left, and lines end with `ESC[K`.

### 6.6 Missing commands

`MATH C_ADD`, `FLUSH`, `MM.INFO(FREE SPACE)`, `OPTION CRLF`, `OPTION PRINTSCREEN`, `PLAY BBC SOUND`, `GUI CURSOR`, `DEVICE(MOUSE ...)` and `WEB ...` do not exist. A short loop replaces `MATH C_ADD`; on the PC this costs nothing measurable. According to the source MMB4W does not call `SUB MM.END` at the end of a program; MMbg therefore cleans up itself before every `END`.

### 6.7 Fan and CPU load

MMbg waited for keys in a loop without a pause. On the PicoMite nobody minds, on Windows it keeps one CPU core at 100 % – the fan speeds up. Measured with a test program: **49.7 million** loops in 20 s without a pause, **2000** with `PAUSE 10`, load hardly visible. All loops waiting for keys now use `PAUSE 10`. There is no waiting while the computer thinks, so the game does not get slower.

### 6.8 Clipboard via PowerShell

MMB4W gives programs no access to the clipboard (only its editor has it). But there is `SYSTEM command$, a$`: the command runs in Windows and its output lands in the string. With it MMbg fetches the clipboard on Ctrl+V:

```
SYSTEM "powershell -NoProfile -Command Get-Clipboard", a$
```

Measured: 769 ms (PowerShell start). The first line is taken, printable characters only.

### 6.9 gnubg over TCP – without a timeout

Measured with a network test against gnubg 1.08.003 using the network address of the gnubg PC (not localhost):

| Case | Result |
|---|---|
| gnubg waiting | connected in 0 ms, answer to `version` in 0 ms |
| 10 × ping | 10 of 10, average 0.2 ms, max 0.4 ms |
| gnubg not waiting | "connect failed" after about 2 s, can be trapped |
| `TCP RECEIVE` without data | **waits for ever**, `MMBasic.exe` must be restarted |

MMbg therefore never waits for more lines than gnubg will surely send: a one-line "Error" answer ends the reading, an empty block means "connection closed" and leads to the fallback engine. If gnubg itself hangs, MMbg waits with it – there is no way around that from MMBasic. A real timeout would need an extension of `TCP RECEIVE` in MMB4W.

### 6.10 Sound without extra files

`PLAY SOUND channel, B, N, hold, volume` produces white noise. For type `N` the "frequency" in MMB4W is the **hold time** in samples (44.1 kHz): 1 = brightest noise, 32 = very dark. The volume goes from 0 to 25 per channel. A `SETTICK` interrupt (nr 4) rebuilds the envelopes of the BBC sounds every 10 ms. The values were tuned by ear with a test program:

| Sound | Hold | Volume |
|---|---|---|
| Dice | 3 | 1.00 |
| Slide | 8 | 0.12 (random flutter, fading) |
| Landing | 1 | 0.95 |
| Hit | 1 | 0.80 |
| Tray | 8 | 0.75 |

Two lessons: a new sound must not replace a short click that is still playing – otherwise the landing was swallowed by the slide of the next partial move and only a constant hiss was left. The slide is now queued behind it. And there is no `ON ERROR` in the interrupt, because it could change `MM.ERRNO` in the main program.

## 7 Measurements

Measured by Matthias on an AMD Ryzen 7 5700G under Windows 11 Pro with MMbg Win (development builds w0.6 and later, engine code as in 1.10.36).

| Run | Result |
|---|---|
| Loading | small nets 26 ms, big nets 15 ms |
| C4 big vs big | 0.98 h, 258 matches, **13.7 s per match**, move list median 53 ms, 0 errors |
| C4 big vs gnubg 2-ply | 56 matches, big 26 : 30 = 46.4 % (± 13 %), 13.7 s per match, gnubg answer median 21 ms |
| C4 big vs gnubg 0-ply, duplicate | 2.7 h, 831 matches, 2999 games, **big 47.3 % (± 3.4 %)**, 11.8 s per match, gnubg answer median < 1 ms, 0 errors |

For comparison on the PC3: big vs gnubg 2-ply 46.7 % from 1236 matches, about 6.5 min per match. On Windows the same amount of data is produced about 28 times faster. Against 0-ply, 50 % is just inside the uncertainty range – the statement "plays about like gnubg 0-ply" holds, with a tendency to slightly weaker. Notable: the points are almost equal (3278 : 3252), but big wins more gammons (316 : 257) and backgammons (53 : 24), gnubg more single games (706 : 599).

## 8 Limits and open points

- No timeout for gnubg answers (see 6.9).
- The CMD window is display only; input only in the graphics window.

- Move list (`L`) and dump (`D`) appear in the CMD window without positioning.
- The sound is a rebuild, not a BBC sound; default off.
- No screenshot key, no free-space check for the log.
- Only tested with MMB4W V5.07.03b25 under Windows 11 Pro.

## 9 Licence and taking part

MMbg is free software under the **GNU General Public License, version 3 or later**. The weights of the neural nets come from GNU Backgammon and are also under the GPL. The source is open: every line of `mmbg_win.bas` is readable MMBasic.

That is exactly what MMbg invites you to do: look inside, understand, change, improve. On Windows this is especially easy – open the program in the editor, try something, run it again. Whether you are a student who wants to know how a computer learns to "think" in backgammon, a hobbyist or an experienced programmer: improvements, bug reports and success stories are welcome on TheBackShed forum.

Have fun playing, trying things out and improving – and with your next roll.

*Matthias Hollatz*

