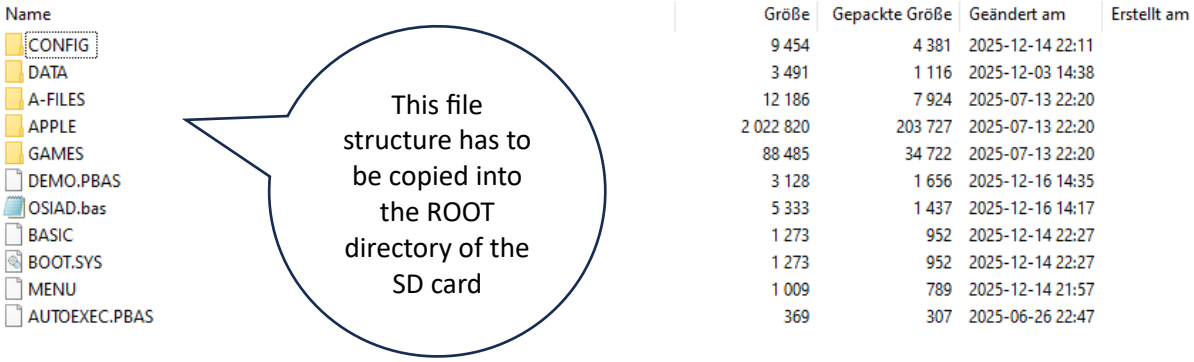
QUICK GUIDE

SD-CARD EXTENDED BASIC V8

1 Software

- Make sure the SD card is formatted in FAT32 and standard cluster size of 16kB SD card size is limited to 32GB.
- Copy the System Boot and Games files to the SD-card
- OSI.SD (aka ZIP) contains these system files for different machines:
  8K, 16K 24K,32k and 40k RAM machines. Choose the directory that fits to your machine memory layout!!



- BOOT.SYS is the file executed on BOOT (by LOADER.SYS)
- LOADER.SYS is the file loaded by the monitor “L” command
  These files can be modified to your needs but must keep the same filename.

2 How to Boot into the SD card?

Starting BOOT.SYS you have to do the following

1 First you need to do a cold start “C” to initialize BASIC and Memory
(this is only necessary at first power on to initialize BASIC)

2 Press RESET again

3 Press “M” for machine code

4 Press “L” within 2 seconds after reset. This is the time, where the SD card is waiting for you. To repeat the autoboot feature, start at 2 again. If the “Loading…” screen does not appear, the SD card was not recognized or BASIC is not initialized.
NOTE: Tested on SYN600 and CEGMON ROM’s. Other ROM’s may not work.

- Check SD card format, max size of mini.SD card is 32 GB
  in FAT32 formatted
- Check if the correct machine files have been used (8k ... or 40k versions)

- Check correct SD card insertion
- Check mechanical connector setup

The system will boot into the BOOOT.SYS file on the SD card. BOOT.SYS is a program that provides extended BASIC commands. BOOT.SYS can be any alternative program of your choice. BOOT.SYS is in this version identical to the program “BASIC” and will show the usual BASIC “OK” prompt. It provides extended resident BASIC commands.
BOOT.SYS will occupy 1280 bytes at the TOP of RAM and adjust the BASIC memory pointers accordingly. It runs well on 8k machines. It also will place a jump instruction at $00BC to process the extended BASIC commands. LOADER.SYS is a typical OSI loader file for the Monitor and it will start BOOT.SYS if present in the root directory. With a modified system ROM, it is possible to load BOOR.SYS by the “D” startup command instead of the OSI disk system. Important: The SD card is not compatible with OS65D disk operating system, as this software requires the hardware of a 610-disk controller. Although a modified OS65D may be possible in the future and make use of the virtual disk capability.

Using the File MENU program

Before explaining the new BASIC commands, you can run a program called “MENU” on the SD-card to browse the filesystem and run or load programs from there.
To quickly enter into the MENU program, type the command **SDRUN**. The MENU program is located 768 bytes before the extended BASIC program and does not stay resident.

Navigation is done by the three keys
“ >” , “<” and “/” on the OSI keyboard

“ESC” will go back to BASIC with extended BASIC available.
Key “D” will delete the selected file without warning.

UK101 users:
Instead of ESC, you can type
Shift-Control -K

“ENTER” will allow to
A) go into a subdirectory
B) load and/or start a program

“[]” indicates a subdirectory

“/” will bring you back to the root directory

Pressing ENTER on a .PRG, PBAS. .LOD, 65V, HEX type of file will load and start the file.
.BAS are Basic text files .LOD and .HEX and .65V are OSI machine loader text files .PRG or PBAS are binary program files

3 Extended BASIC for SD card

In this BASIC extension, PATH and FILENAME are separated. Once a PATH is set, all file operations are done within the path given.
Here the new BASIC commands available:

SDLIST	List current directory
SDLOAD “FILENAME”, [ADDRESS]	Loads BASIC or Binary file
VAL=SDSAVE “FILENAME”, [START ADDRESS, END ADDRESS]	Saves BASIC or binary file
VAL=SDLEN “FILENAME”	Returns length of file
SDGOSUB “PATH”	Select Directory
VAL=SDLOG	Returns last error
VAL=SDCLEAR “FILENAME”	Removes a file
STRING=SDCHR$ (x)	Get directory entry at position x “” (empty) returned if nothing found
VAL=SDFRE	Returns free space on SD card
VAL=SDVAL	Validate SD / Close file, return 1 if OK
SDDEF 0 or 1	Set/Reset AutoStart for BASIC files
SDRUN	Runs the “MENU” program

BASIC commands only available <u>on 16k RAM or more</u>:

VAL=SDNULL ”DIRECTORY”	Remove directory in current path.
VAL=SDDIM ”DIRECTORY”	Create directory name in current path
VAL= SDASC “FILENAME”, “FILENAME”	Rename File

On commands returning a VAL, it is not needed to do an assignment. For example, VAL=SDCLEAR “FILENAME” will work as well as SDCLEAR “FILENAME”. Here VAL returns a “0” if the command failed and a “1” if succeeded.
PATH string length is max 31 characters; FILENAME string length is max 23 characters.
FILESIZE is max 65535 bytes for saving. Read files may be larger. SDLEN returns 65535 in length, if file is larger than 64k or zero, if a file does not exist.

INFO:

SDVAL may be used like a CLOSE command after reading or writing files.
I case the SD commands get stuck due to program interruption or mistaken commands, SDVAL will reset any SD card error and you can continue operation.

SDLIST will show a directory listing of the actual path. The listing may be interrupted by pressing the right "SHIFT" key. Changing the path can be done by **SDGOSUB** "Path".
Type SDGOSUB "" (empty string) to return to the root directory.
Several path levels are separated by "/", for example SDGOSUB "APPLE/TEST"

Programs are always saved in binary format, even standard BASIC programs.
To save a BASIC program under extended BASIC, use the *.PBAS extension.
A file with a .BAS extension is the conventional text file format and will be loaded like a file form the serial/cassette port. Under extended BASIC, you **cannot** save Basic programs in text file format (for now). But loading of old text file formatted software just works fine.

If you want to save your BASIC program as an AutoStart program, enter **SDDEF** 1.
Do this before save. SDDEF is default 0. For example, SDDEF 1: SDSAVE "TEST.PBAS"

Without an address field, the current BASIC program will be taken and saved in binary PRG encoded format (BASIC binary includes a specific header with starting @ 02D8…).
You may chain AutoStart basic programs by SDLOAD within a BASIC program. Variables will be cleared. For transfer of data, you need a dedicated memory area.

If you add the optional [ADDRESS START, END] field, a machine code PRG file will be created with AutoStart at the given START ADDRESS.
If you use **SDLOAD** "FILENAME" the file will be loaded at its original location and executed, if the extension starts with *.P…. Otherwise, the SD command just returns.
Means saving and loading a data file *.DAT by a given start and end address allows you to store and retrieve binary data to a specific location without executing.

If you use **SDLOAD** "FILENAME", ADDRESS, the file will be loaded to the given address instead. Important: If you use a target address, the original start address in the file is ignored. See EXAMPLE 1 here and in folder CONFIG.

A value returned (data or error status) indicates, if the SDSAVE, SDREM failed (0) or passed (1). To check if a file exists, use **SDLEN** "FILENAME". A return value >0 shows the file size and that the file was exists. (see also SD_Card_Reference.pdf)

IMPORTANT:

The file extensions are relevant to determine the type of program code. Upper and lower case letters as file name or extensions make no difference. We have:

*.PRG, *PBAS, * (no extension), all file extension starting with “P “or empty

-> These are binary AutoStart executables with a leading load/start address

*.BAS, or all file extension starting with “B “

-> These are true BASIC text files that will be loaded by the BASIC interpreter

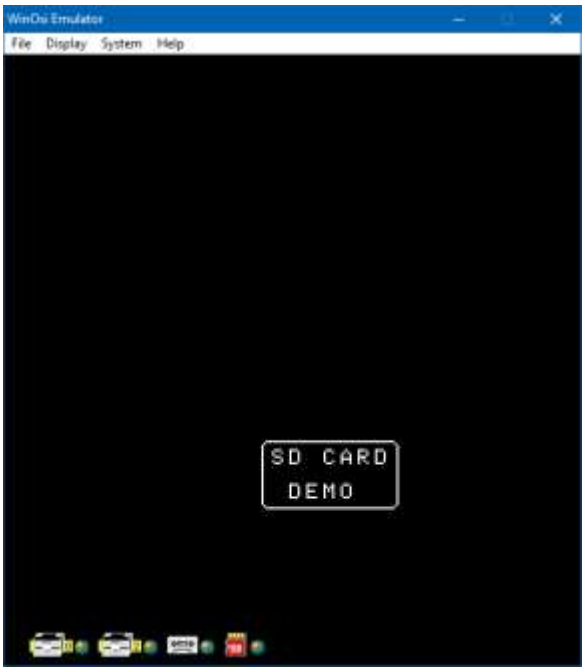
*.LOD, *.HEX, *.65V or all file extension starting with “L “ , “H” or “6”

-> These are OSI Hex loader text files that will be loaded by the OSI MONITOR

All other file extensions are handled like binary executables with a leading load/start address but these will not AutoStart.

Example 1:

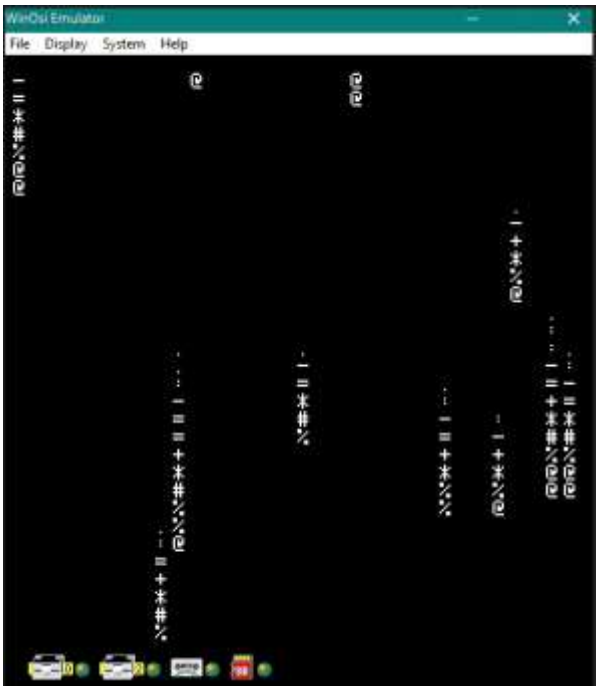
```
10 REM BOING DEMO
20 SDVAL: SDGOSUB"CONFIG"
30 QB=SDLEN"DEMO.IMG": IFQB=0 THEN
    PRINT "FILE NOT FOUND": END
40 FORQA=0TO30:PRINT: NEXT
50 PX=0:PY=0: DX=1: DY=1
60 PX=PX+DX:PY=PY+DY
70 IF PX>7 THEN DX=-1
75 IF PY>20 THEN DY=-1
80 IF PX<-7 THEN DX=1
85 IF PY<2 THEN DY=1
90 SDLOAD"DEMO.IMG",53248+PX+32*PY
100 FOR QA=0TO31: NEXT: GOTO60
```



Here a DEMO.IMG binary file (with a leading load/start address) is placed at different positions on the screen, ignoring the original load address.

Example 2:

```
10 REM READ DATA IMAGE DEMO NEW
20 FOR QA=0TO27:PRINT:NEXT
30 A$="ANI.IMG":B$=A$+"01"
40 A=7+PEEK(129)+PEEK(130)*256
50 SDVAL: SDGOSUB"DATA"
60 QB=SDLEN B$:IFQB=0 THEN PRINT
   "FILE NOT FOUND":END
100 FOR QC=1 TO 90
110 IF QC<10 THEN QB=48:GOTO 130
120 QB=48+INT(QC/10)
130 FOR QA=0TO20:NEXT
140 POKE A,QB:POKE A+1,48+QC-
    (INT(QC/10)*10)
150 SDLOAD B$
160 NEXT
170 GOTO 100
```



Here a listing of 90 animation images loaded into screen memory.

Several other BASIC files are on the SD card to demonstrate the functions. GETFILE.BAS shows how to access the SD card without the BASIC extension present. Directly from standard OSI BASIC (if you want to access the SD card without the BASIC extension). **Using the BASIC command extension is much more effective/faster!**

REMARK: When running the SD MENU program, the BASIC command extension will stay resident. MENU uses some routines in the same memory space at the end of memory. The MENU program and other programs should be placed below in memory, to keep the BASIC extension available.

On 8K machines, some programs occupying more than 6kb program space may not work. Here you need at least a 16kB machine.

Just returning to BASIC can be done by pressing the “ESC” key. The BASIC command extension is kept in memory. UK101 users, you have to type Shift-Control -K
Basic programs in memory will not be changed and can be edited or run normally.

If you once lost the BASIC extension (SD… commands report a Syntax error), hit the BREAK key, and do M and L to reload.