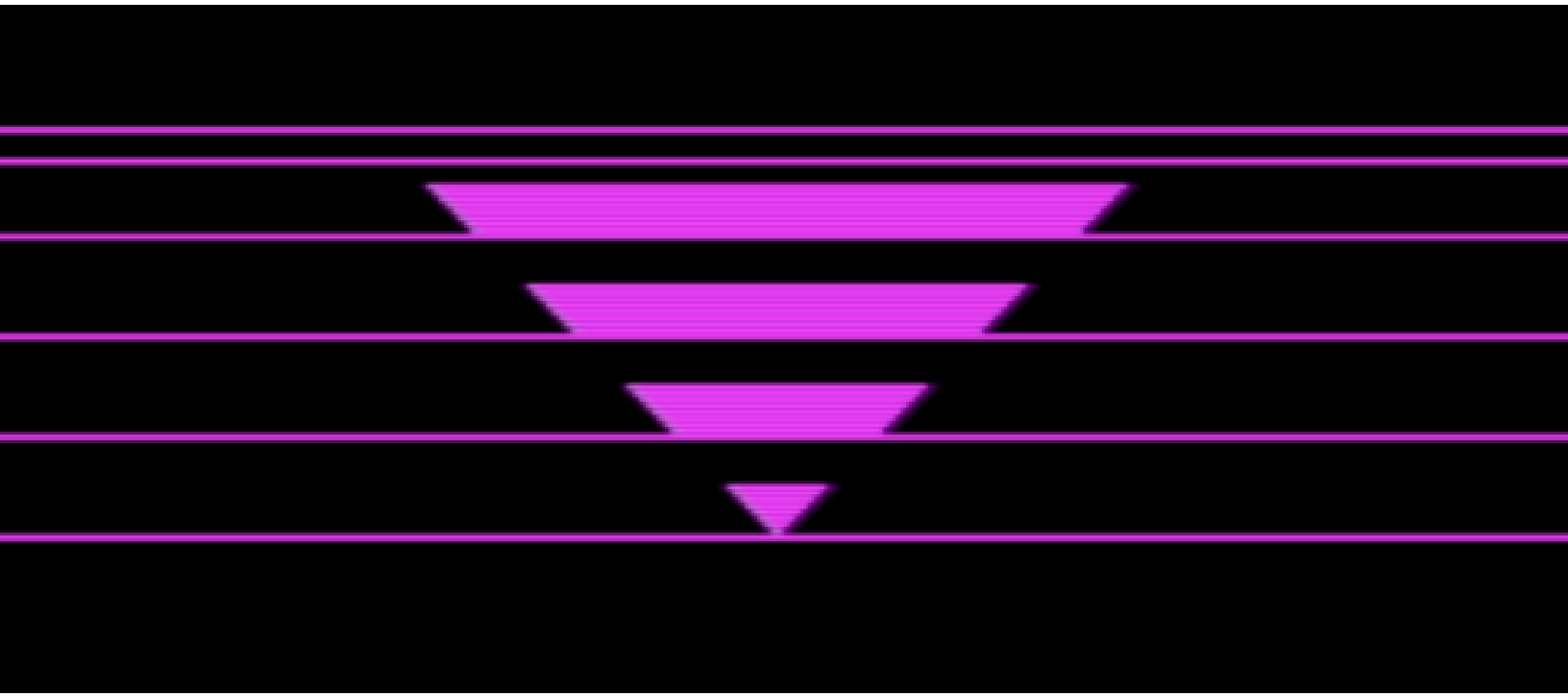

TRIANGULAR µOS 1.28

for

Commodore CBM-II

Programmer's Reference Guide

Contents:

A. Introduction

Programmer's Reference Guide for TRIANGULAR µOS 1.28 SDK (Software Development Kit) explains programming aspects of TRIANGULAR µOS, a GUI (graphic user interface) operating system for 8-bit Commodore computers.

Goal of creating this system was to develop GUI for 8-bit Commodore computers with lowest amount of memory: that is Commodore PET with at least 4 KB of memory. Next it was expanded for Commodore VIC-20, Commodore 64 and Commodore 128. This iteration (version 1.28) of TRIANGULAR µOS requires Commodore CBM-II.

This software was written in Commodore BASIC language (port of Microsoft BASIC) using CBM prg Studio 4.1.0, and is designed to run on standard Commodore CBM-II. TRIANGULAR µOS is supports BASIC 4.0 and works in color text mode. Commodore BASIC (a runtime interpreted language) is default language used in 8-bit Commodore computers and also functions as their OS and user interface. Similarly, to early Microsoft Windows (1.0 to 3.11), µOS sits atop of BASIC and KERNAL (Commodore's kernel) and Commodore DOS, which is implemented in every Commodore disk drives or 3rd party solutions in order to load µOS modules, load/save settings and documents, perform operations on floppy disks and communicate with disk drive(s).

Package contains files:

- *TRIANGULAR uOS 1.28 for Commodore CBM-II Programmer's Reference Guide.pdf* – this document
- *Source Code* folder with 3 files: *UOS.bas* (source code of UOS program), *GUI.bas* (source code of GUI program) and *uos.cfg.seq* (default configuration file)
- *TRIANGULAR uOS 1.28 for Commodore CBM-II System Registry.xlsx* – spreadsheet which details usage of variable stored in memory by TRIANGULAR µOS 1.28
- *TRIANGULAR uOS 1.28 for Commodore CBM-II System Disk Content.xlsx* – spreadsheet which presents file structure of System Disk of TRIANGULAR µOS 1.28
- *TRIANGULAR uOS 1.28 for Commodore CBM-II Variables.xlsx* – spreadsheet which details BASIC variables used by TRIANGULAR µOS 1.28
- *TRIANGULAR uOS 1.28.d64* – empty, preformatted System Disk
- *TRIANGULAR uOS 1.28 Documents.d64* – empty, preformatted Documents Disk

B. What you need

In order to change and/or compile TRIANGULAR µOS 1.28 you need to do this using external program like CBM prg Studio 4.1.0 (which was used in development and for compilation of .prg files). Using BASIC 4.0 on real hardware or emulator is out of question, since source code uses extensive line concatenation (up to 255 bytes long lines). Standard BASIC won't present program lines properly (especially print statements) and its screen/program editor won't be able alter those lines.
Download CBM prg Studio here: www.ajordison.co.uk

For fast creation and modification disk content I recommend DirMaster. I formatted my disk with custom PETSCII characters in Disk name and Disk ID in TRIANGULAR µOS CMD program (N>TRIANGULARµOS<µ - µ symbol can be achieved by pressing C= + M and then C= + X).
Download DirMaster here: style64.org/dirmaster

For testing and debugging use real Commodore CBM-II or emulator (I use freeware VICE emulator).
Download VICE emulator here: vice-emu.sourceforge.io

Commodore CBM-II emulator VICE must be configured with enabled disk drive that can read 170KB 5.25¨ diskette (.d64 file): recommended CBM 2031 (default). Also, you should enable joystick in Port 1. You can easily configure it as Numpad keys:

- Up (8), Down (2), Left (4), Right (6)
- You can move diagonally e.g., Up-Left (7)
- 0 or right Ctrl: Fire (click/select)

You can also enable printer in in VICE emulator. Do this in:
Settings -> Peripheral devices -> Printers or similar options. You can choose printer as device #4 - #7, although #4 is standard and recommended.

C. How to compile TRIANGULAR µOS 1.28

Source code of UOS and GUI programs is stored in UOS.bas and GUI. files. Segments of programs are commented with simple descriptive caption-like comments (!- is used to comment out line in CBM prg Studio).

After compiling source code files add UOS.prg and GUI.prg program files to System Disk. File names on disk should always be UOS and GUI (in upper case/graphic mode) or uos and gui (in lower case mode). Remember to put UOS file first (to properly load system with LOAD "*",8 command).

Add uos.cfg.seq file and change its name to UOS.CFG (it should have SEQ property) and place it in the middle of UOS and GUI (that's my convention).

You can use empty, preformatted System Disk file to speed up process (TRIANGULAR uOS 1.28.d64 file).

D. Troubleshooting

Loading of next module of TRIANGULAR μOS can “freeze” in process of inter-loading next μOS module or disk program (very rare occurrence). This happens when loading screen not proceeds to next module for over 2 minutes for μOS (smaller program will take less time to do so). When loading screen is not responsive for longer time, it means error in inter-loading procedure, most probably keyboard buffer was not filled with key properly. To see what really happened change color of cursor to blue (press Control + 7) and enter command POKE 55329,1 and hit Return key. This should change background color to white which will show underlying black text of loading sequence. If computer doesn’t change cursor or background color try again. If still there is no effect it might be real freeze. If color change procedure succeeds, try using RUN command to see if program will start or go to top of screen (Home key) and press Return in order try to reload program. If it will loads successfully enter RUN command. If that not work check if load command is correct. It should have format: LOAD “[filename]”, [device # (1 or 8 - 11)] like in e.g.: LOAD “GUI”, 8. If none of it works then start system anew. To prevent this kind of freeze, try not to use keyboard when inter-loading procedure is performed (it can slip improper key into keyboard buffer, which most often leads to this error).

E. BASICALLY

Below are listed functions of nascent BASICALLY API of TRANGULAR μOS 1.28.

1. Window generator: draws empty window based on data in variable arguments. Before evoking this function assign desired values to those variables:
   w1 - window top-left vertical position
   w2 - window top-left horizontal position
   w3 - window bottom-right vertical position
   w4- window bottom-right horizontal position
   wn$ - window name which will be displayed on title bar and on task bar

   Next evoke this function with gosub51 command.

   This function also generates w1$ string variable (vertical start of window) and w2$ (horizontal start of window) that can be used in further alignment of windows elements Caution: j variable is used in for-to:next loops.

2. left$(s$,x) – will display x number of spaces (max x = 39)

3. left$(v$,x) – will display x number of cursor down {down} (max x = 23)

4. left$(h$,x) – will display x number of cursor right {right} (max x = 39)

F. Support & Legal note

More information about TRIANGULAR µOS for Commodore CBM-II or other computers system is available on TRIANGULAR µOS website, where you can download SDK, report bug or get help: triangular-uos.blogspot.com

LEGAL NOTE:

G. Changelog

TRIANGULAR μOS 1.28 for Commodore CBM-II [29-10-2023]:

- Supports Commodore CBM-II P/500 series
- GUI: PETSCII mouse pointer reintroduced and Start Menu button is sprite-less
- SNAKES and LUNAR LAND from PET version in GAMES folder in place of removed TAXI, RIFT RESCUE and CHUCK CHALLENGE games
- Minor bugfixes and improvements