DOS/65 Version 3.20 Release: 4th April, 2023
By Kevin E. Maier

This release is based on DOS/65 Version 3.0, written by Richard A. Leary. There is still large core of Richard's original code in this release, but there's a large number of changes, including certain routines which have been replaced, re-written and in some cases, removed. There are also major changes in the way disk drives are accessed.

Richard's code was essentially a port of CP/M from the Intel 8080 over to the 6502 (original NMOS CPU). The intent was to provide a standard CP/M-like operating environment on the 6502 that had a common filesystem format and system calls. While this would not provide a binary level of compatibility with CP/M, it would provide a common code structure and compatibility at a disk level.

DOS/65 has been licensed code directly from Richard for many years. The last version that Richard worked on (to the best of my knowledge) was Version 3.0. There are two versions of the 3.0 code: A RAM based, bootable version (using SYSGEN) and a ROM based version. These final versions did provide some additional features and functions beyond the previous 2.1 version, which are highlighted below:

- *A CP/M compatible User area for storage devices is now supported*
- *A CP/M compatible SUBMIT batch processing capability is added to CCM*
- *The IOSTAT variable now provides the USER number and Drive number*

There are some functional limitations of DOS/65, which are highlighted below:

- *A maximum of 8 drives (drive letters A – H) are supported*
- *Each allocated drive is limited to 8MB total size*
- *A single file has a maximum size of 512KB*

In the DOS65 V3.0A Supplement Rev A (March 2021), Richard changed the license agreement for the 3.0 ROM Version as shown below:

"The DOS/65 V3 ROM software and all other DOS/65 software are free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the license, or any later version."

My (personal) view is that Richard has stopped supporting and updating DOS/65, hence the change in the licensing to GNU GPL Version 3 or later. By referencing "all other DOS/65 software", this *should* include earlier releases and the accompanying utility programs, most of which were released under Version 2.x . This would include applications like the Assembler, Basic Compiler, Runtime for the Compiler, Editor, etc. If anyone has knowledge that this is not the case, please let me know by email (address below). My last correspondence with Richard was around the end of 2021. Hans Otten did exchange some correspondence with him in 2022. Based on this, Hans has provided some details and links on his website for DOS/65 here:

http://retro.hansotten.nl/6502-sbc/dos-65/

There have also been multiple threads and discussion on 6502.ORG and some other sites which Richard participated in directly for older systems such as the OSI, which can be found here:

http://osiweb.org/osiforum/viewtopic.php?f=4&t=235&start=40

I started working with DOS/65 back in 2021 and have made numerous changes and updates to CCM, PEM and modified a SIM module to interface to the BIOS of my C02 Pocket SBC with the Compact-Flash and RTC adapter. This was a ROM based version and was smaller and a bit faster than the original ROM code from Richard.

I started working on a RAM based version which is bootable from disk in late 2022, which would allow for a larger RAM space and reduced ROM footprint. There are many significant changes with this new version. Some of these are highlighted below:

- *Separation of CCM, PEM and SIM into standalone modules*
- *Rewriting CCM/PEM modules to take advantage of CMOS opcodes and addressing modes*
- *Fixing a few problem areas for drive letters and out of range inputs to CCM/PEM*
- *Elimination of Track/Sector based disk accesses – all disk access is Record based*
- *Elimination of Checksum routines in PEM, used for diskette-based media*
- *Elimination of all code that supports removable media (e.g., diskettes)*
- *Update to the Disk Control Block (DCB) for Record Based media*
- *A completely new SIM module with enhanced blocking/de-blocking*
- *Source code ported to WDC Tools for assembly/linking*
- *Version 2.1 utilities are still fully compatible (Assembler, Editor, Basic, etc.)*
- *Updates to some core utilities such as SD to accommodate the updated DCB (more in the works)*

The above work has resulted in smaller CCM and PEM modules, which can run on any CMOS processor. Many of the changes to PEM calls are specific to disk-based access routines. Using the updated DCB, Record calls to SIM also include Reserved Records per the DCB which are passed over to SIM. The Reserved Tracks call to SIM is no longer used, as PEM will add the offset for Reserved Records per drive when calling SIM. Also note that the PEM call to translate sectors (diskette only) is now considered obsolete and simply returns, doing nothing. The Home Drive SIM call is also obsolete due to non-removable media. The DCB has been modified slightly to reflect Reserved Records, not Tracks, and eliminates the last 3 bytes which define the checksum enable byte and the checksum map address. These calls are no longer necessary as supported storage devices are all LBA based and non-removable. This can include any standard media such as IDE, Compact Flash (in IDE mode), SCSI block devices and newer devices such as SD-Card. In short, any media that can be interfaced to a 65C02 system and addressed via LBA access can be supported with a compatible BIOS that SIM can interface with.

With rare exception, any existing application for DOS/65 should run without changes. To date, I've assembled and tested almost every utility that was provided with DOS/65 Version 2.1 and 3.0. Of course, there are certain applications/utilities that are specific to the hardware platform, such as DEBUG, that need to be customized for the hardware being used. As my particular system already has a high function Monitor, I've opted to not make changes to the DOS/65 DEBUG program. I also have no plans on updating it, as I don't need it for my 65C02 system(s). If anyone else chooses to use this new DOS/65 version (3.20), they might want to consider updating the DEBUG program to use with their system.

The hardware system I have used to develop and test the new version on is my C02 Pocket SBC prototype system, which is described below:

- *W65C02 Processor running at 8MHz*
- *AT28BV256 EEPROM addressed from $E000 - $FFFF (8KB)*
- *Alliance 128KB SRAM addressed from $0000 - $DFFF (56KB)*
- *I/O Window configured at $FE00 – 5 I/O selects, each at 32-bytes wide*
- *NXP SC28L92 DUART – both ports and timer configured*
- *Maxim DS15x1 RTC clock (not used with DOS/65 currently)*
- *Hitachi 6GB Microdrive PATA ZIF with 16-bit IDE interface*
- *TL7733 Reset Supervisor and DS1233A for NMI Panic trigger*
- *BIOS supports DUART and Microdrive via interrupt-driven code*
- *BIOS Jump table at $FF00 is called by SIM for hardware access*
- *All hardware is now 3.3 Volt only (Microdrive is only 3.3V power)*

DOS/65 Version 3.20 requires 5KB of memory for CCM, PEM and their associated Data areas. The SIM module is mapped to 1KB of memory after PEM, but currently only uses 768 bytes. All modules are mapped to page boundaries. There are 9 pages for CCM, 11 pages for PEM and 4 pages allocated for SIM. This totals to 6KB of RAM and the default storage configuration provides for eight drives, each at 8MB size. The drives are configured as:

- *4096 total blocks*
- *2048 bytes block size*
- *1024 directory entries per drive*
- *4 Records per Disk Block*

The additional 4KB for allocation maps brings the total DOS/65 size to 10KB. Note that the C02 Pocket SBC prototype uses the first 2KB for hardware support: Page Zero, Stack, Software Vectors, Serial and Disk buffers. The upper 8KB is mapped as ROM and contains a rich function Monitor, the BIOS and the I/O mapping. Loading from the Reserved Records area of the first drive, DOS/65 is loaded starting at $B800 and the TEA starts at $0800. This provides 44KB of free memory for user programs, with an additional 2.25KB by overwriting CCM if required.

It is possible to reduce the ROM size by shrinking or eliminating the Monitor code, which could increase RAM by roughly 6KB. Note however that the Monitor does have some useful code which can help with debugging and benchmarking. It also allows full examination and modification of memory, including a disassembler and a XMODEM-CRC utility with S19 record support. Currently, the C02 BIOS allocates the upper 48 bytes of Page Zero (2 bytes are free) and the C02 Monitor allocates 48 bytes of Page Zero directly below the BIOS (1 byte is free). DOS/65 Version 3.20, which is RAM based only uses 8 bytes of Page Zero space. This provides 152 bytes of contiguous Page Zero space for user applications. It's also possible for user applications to use some of the Monitor's Page Zero space if required.

To date, extensive testing has been done on the current Version 3.20 final release, which includes extensive disk activity with utility applications including:

- *SUBMIT*
- *Assembler*
- *MakeCom*
- *Alloc*
- *SD (Super Directory)*
- *Basic Compiler and Runtime*
- *COPY and UCOPY*
- *COMPARE*
- *DUMP*
- *MORE*
- *PAGE*
- *XMODEM*

The 3.3-volt hardware system has been completely stable without any crashes, running for months non-stop. The 3.20 release level of DOS/65 has also been completely stable without any crashes. The code has been exercised extensively with “sub” files that copy files from drive to drive, compare files from drive to drive, build the entire toolset of utilities, Xmodem uploads/downloads, etc. There simply have been no failures as of yet (all fingers crossed).

CCM and PEM now check for disk drive ranges and show an “UNKNOWN DRIVE” error if the attempted drive is out of the range of the system. Furthermore, there are no longer any possible phantom drive access issues (e.g., trying to access drive t:, or even a non-alpha character), which could cause disk problems if inadvertently used from CCM or a call to PEM. Note however that many of the existing utility programs do not properly test for drive letters that the user can enter. In general, these utilities simply mask off the lower 3-bits for the drive letter. Over time, I will start to work on these utilities so they are updated to properly handle out of range drives. Other application/utility issues are a lack of a consistent exit to DOS when finished. Some applications/utilities return to DOS from a RTS, while others exit by a Warm Boot to PEM, which is preferred.

The new SIM module manages blocking and de-blocking for reading and writing blocks to the disk. SIM keeps track of the current 512-byte disk block buffer and matches it to the Record number requested by PEM for reading and writing. SIM also provides a performance enhancement to Record reads and writes. A single disk flag byte provides separate flags for read and write operations in progress, a valid LBA loaded, with a Dirty Block flag for delayed writes and the 2-bit Record offset within the currently loaded LBA. This has resulted in very good overall performance and a smaller SIM memory footprint. Note that PEM no longer needs, or uses, a “write type” when calling SIM for disk writes and the associated (PEM) code has been removed.

With Version 3.20, there is no SYSGEN level of system creation. WDC Tools are used for assembling CCM, PEM and SIM, with the linked output as a Motorola S-Record (S19) file. The C02 Monitor supports downloading via XMODEM-CRC of S-records, which can quickly download the complete DOS/65 Image. I’ve written a disk utility that provides a fair amount of function for testing and examining an IDE device. A recently added function is to write a section of memory to a contiguous set of blocks on the IDE device. As the C02 Pocket SBC is using a disk partitioning scheme to provide multiple bootable partitions, a partition offset is added to SIM which defines the starting LBA for all of its drives. The actual utilities to setup the storage device and configure the partition and boot records is still a work in progress. For the time being, the C02 Monitor simply loads the image from the drive to memory and jumps to the cold-start address for SIM. The warm-boot code in SIM reloads CCM and PEM from the reserved section of the first drive.

Any system which has a bootable disk feature should be able to boot this version of DOS/65, but will require some minimal changes to the SIM module to interface to the local machine BIOS.

I’m hoping some others can take advantage of this new version of DOS/65 and get it running on a wider range of 65C02 systems. The only functions required to implement this are a compatible BIOS to support the console and disk-based calls. This has been a fun project and many thanks to Richard for his time and effort to provide DOS/65 to the 6502 community.

Best Regards, KM
kemaier2@bellsouth.net
https://github.com/floobydust