Rock Crush And Dans Revenge - The Definitive Edition Release: Sept 2018.

Introduction:

During the early to mid 1990's, when emulation began to take off, I wrote some tape utilities to enable me to transfer my games over from cassette to the PC.
Like most people, my real ZX81 was retired up the loft and was no use in transferring data over to a PC anyway.

Once the games were rescued, I noticed that they didn't run exactly as they used to on the emulators - mainly Carlo Delhez's (great for the time), Xtender program.
They either ran too fast or too slow or just had a strange feel to the gameplay.

I made a few internet releases during this time: V1, V2, V2.1, V3 and the unreleased V4.

Each of these releases either played with the timing values used in the games, patched the machine code to wait for a certain amount of FRAMES to pass or a combination of both.
Some even tried to detect Xtender and modify the program delays accordingly.

Hindsight is a great thing but at the time, I just wanted the games to be playable and enjoyed…

Fast forward to 2018 and I want to officially retire all those releases and replace them with the 2018 Definitive Editions!

Time has moved on and modern emulators are more than able to run the games just like they did on a real ZX81.

These versions here therefore, are as close as possible to the original tape duplication masters as you can get - they were loaded into a real ZX81 from this actual tape and saved off using my ZXpand interface board.

Timings and delays are as originally written for the UK market 50hz ZX81.

There is however one big change I have made this time around.
It has taken me over 33 years but finally I have sat down and worked out a way to get the game to load in one part!

I gave myself one design goal however.. Should the user manage to break into the program before the game starts up (by pressing "S" or "Q" keys for example - both do slightly different things!), they should be able to still operate the BASIC environment, look about and find the state of the machine (when pressing "S") to be identical to holding down the "S" key on the original 5-part loader.

I will use this facility to make a small cassette release of the games and am currently working on a CHROMA 81 colour version of Rock Crush.

More on that will follow later!

It has a been a long time since I wrote Z80 assembly code but I find that part easy - harder is getting your mind around WHY things were done a certain way back in the day and the state the machine was in during development with regards to the memory map. It took a bit of digging about and disassembly work to get back into the way things were but I am enjoying it now it's all came back to me!

Technical Details of the new one piece loader routine are below (Design Revision 4. Sep 2018):

Overview:

My original games loaded in 5 parts. Part one lowered RAMTOP to 23843 on the fly and the next 3 parts chain loaded each other, block loading code and data above RAMTOP.
The last part contained the main game code and started up the game. The reason for this was simple - not enough room in memory at once to hold the game with source code, assembler above RAMTOP, object code etc. during original development.

It would be nice to get all the original code into the same place in memory as the originals, whilst enabling a one part load...

Unfortunately a lot of heavy memory juggling is required to pull this off and I never got around to tackling it myself until now.

This 122 byte relocatable routine along with some code storage arrays, will accomplish this.

Historical note: Fred Nachbaur did do a one piece loader (with my permission) in the late 80's. He said this was easier for his Disk Drive system! This version is very rare and unofficial as such but I am aware that copies do still exist.
I don't have access to the method he used, nor any details of its structure. It may have even been a relocation as Fred liked and was good at doing this. Look for his unofficial HI-Z version of the HOT-Z assembler by Ray Kinglsey for instance. I have been unable to obtain, break into or view any code he used so was forced to work out my own way which enabled me to set my own design goals as mentioned above anyway.

Here is my method:

The last part 5 (CODE.P) was used as it contains all the main game code. All the data that belongs above RAMTOP is stored in variable B$. A new machine code routine held in A$ manages moving all of this data around in memory and restoring the state of the ZX81 system, as far as the BASIC ROM is concerned etc.

Firstly, the following commands are typed in to prepare the VARS area:

```
CLEAR
DIM A$(122)
DIM B$(8925)
```

The great Eighty One emulator is then used to block load the appropriate code and data into those arrays. (Prepared earlier).

Secondly, the following command is typed in immediate mode to collapse the Display File and allow for faster tape saving:

```
POKE 16389,0 (Fools system once Newline is pressed)
```

Next, GOTO 10 (not RUN) is used to start things:

The BASIC part of Rock Crush now saves the program before calling the machine code routine detailed below, as long as “S” key not pressed, which harmlessly does not execute the machine code :

```
10 REM ***ROCK CRUSH*** - THE DEFINITIVE VERSION
20 SAVE “CRUSH1”
30 POKE 16389,128
40 IF INKEY$<>”S” THEN RAND USR 18502
```

Note: 18502 points in this case to the first chr in A$ {Formula: PEEK 16400+256*PEEK 16401+6}

Line 40 is there in case you do press the “S” key - it takes a minute or so for the BASIC system to respond and be useable again - this being due to the massive arrays being held and the display file needing expanded - the ROM’s method for doing this isn’t very fast to say the least! This at least allows for duplicating tape copies!

Preset variables are A$ (122 bytes) and B$ (8925 Bytes).

A$ is first variable and contains a totally relocatable machine code routine for:

Ensuring the NMI generator is switched off. The stack pointer area will be overwritten with routine below so it is essential no interrupts or stack operations occur.

A block move of machine code held in B$ to top of 16K Ram downwards using LDDR is made. Start address of B$ is calculated from fixed offset from the VARS area.

Once code is moved into place , the new Ramtop value is set and stack pointer is set up along with ERR_SP, below this. Some default values are stored on the stack for the return to BASIC should it ever occur.

To next expand the display file would overwrite this actual routine in memory as it was running, crashing the CPU. The code therefore makes a copy of the rest of itself 1k higher in ram and jumps to this safe copy (resided safely outside of the BASIC program area. The printer buffer area was first considered but ultimately it wasn’t large enough as I needed 54 bytes).

Next code is borrowed from my old “faster new” routine - a new expanded display file is setup . Next variables area VARS is cleared similar to the Rom CLEAR routine.
The NMI generator is then enabled again.

The routine then checks for the “Q” key being pressed using keyboard input port. A return to basic is made if so...
(This is mainly for debugging - it allows for checking the environment in BASIC is identical to the original 5-part loaded Game; which it is!).

Otherwise a jump is made to the original Games entry point - 17737 for Rock Crush or 18012 for Dans Revenge.

B$ contains all data and code that originally resides above RAMTOP.

Some thought was given into storing the code In A$ in its own new machine code REM line but my ultimate goal here was to keep the state of the machine in as similar a setup as the original 5 part loader did. The code and data stored inside the variables area is ultimately deleted before the main game starts up, so is a neater and more accurate solution.

Development Tools Used:

A Mac Pro, iPhone, iPad Pro with pencil, The great Eighty One ZX81 emulator (Windows), ZXPand interface, an Original ZX81, CHROMA 81 interface, Bug Byte’s ZXAS Assembler! DZ80 Disassembler (Windows).

Yes its true.. still using ZXAS on the machine itself.. OK for upto about 4k of code at a time ;-)
I’ll move over to a TASM flat based Assembler one day!

Steven McDonald.

```
;
; Disassembly of the file "CRUSH1.P"
;
; Title:   New Relocatable Memory Manager for Rock Crush / Dans Revenge - 122 Bytes
;
; Purpose: Enables a one piece load by holding all data in arrays which is
;
;          then block moved above a newly set RAMTOP, leaving the machine
;
;          in the same state as the original games did. Variables: A$ holds this
routine.
;
;          B$ holds all data that needs to be moved above RAMTOP. The game is then
started.
```

```
;
;
; Machine: ZX81 16K RAM, Z80 Disassembly.
;
; Developed By: Steven McDonald. Sept 2018.
;


18502 D3FD      OUT     (253),A         ; Switch NMI generator off.
18504 2A1040    LD      HL,(16400)      ; VARS.
18507 116223    LD      DE,9058         ; *Offset to last character in B$.
(6+length of this routine (122)+6+8924=9058).
18510 19        ADD     HL,DE           ; HL = Source.
18511 11FF7F    LD      DE,32767        ; DE = Destination - Top of 16K RAM.
18514 01DD22    LD      BC,8925         ; BC = Length of block.
18517 EDB8      LDDR                    ; Block move - copying Top to Bottom, so as
not to overwrite itself whilst moving.
18519 21235D    LD      HL,23843        ; New RAMTOP value.
18522 220440    LD      (16388),HL      ; Store it.
18525 2B        DEC     HL              ; Back One.
18526 363E      LD      (HL),62         ; Default stack values as expected by BASIC
ROM.
18528 2B        DEC     HL              ; Back One.
18529 3600      LD      (HL),0          ; Default stack values as expected by BASIC
ROM.
18531 2B        DEC     HL              ; Back One.
18532 3606      LD      (HL),6          ; Default stack values as expected by BASIC
ROM.
18534 2B        DEC     HL              ; Back One.
18535 3676      LD      (HL),118        ; Default stack values as expected by BASIC
ROM.
18537 220240    LD      (16386),HL      ; Set ERR_SP.
18540 F9        LD      SP,HL           ; Move SP to point to the newly created
Stack.
18541 2A1040    LD      HL,(16400)      ; Read VARS. Now we are concerned with
copying block of code from L0 to a safe place.
18544 114A00    LD      DE,74           ; *Offset to L0 (6+offset from start of
this routine to L0 (68)=74).
18547 19        ADD     HL,DE           ; Calculate address. Vars+Offset to L0.
18548 EB        EX      DE,HL           ; DE = Source.
18549 2A0C40    LD      HL,(16396)      ; DFILE start.
18552 24        INC     H               ; Add 256.
18553 24        INC     H               ; Add 512 in total.
18554 24        INC     H               ; Add 768 in total.
18555 24        INC     H               ; Add 1024 - now safely 1K higher in RAM.
18556 EB        EX      DE,HL           ; HL = Source, DE = Destination.
18557 013600    LD      BC,54           ; Length of routine L0-End (54 Bytes).
18560 EDB0      LDIR                    ; Move code to safe area.
18562 2A0C40    LD      HL,(16396)      ; DFILE.
18565 24        INC     H               ; Add 256.
```

```
18566 24         INC     H               ; Add 512 in total.
18567 24         INC     H               ; Add 768 in total.
18568 24         INC     H               ; Add 1024 - now safely 1K higher in RAM.
18569 E9         JP      (HL)            ; Jump to safe copy of routine below as
original here is about to be overwritten by DFILE expansion!
18570 2A0C40 L0:LD      HL,(16396)      ; Read DFILE - This part from L0 has been
self-relocated and is now running 1k higher than DFILE in RAM.
18573 3676       LD      (HL),118        ; DFILE starts with a Newline character.
18575 23         INC     HL              ; On one. The display file was collapsed
for SAVING. We are going to expand on the fly - 768 Bytes!
18576 0E18       LD      C,24            ; 24 Lines.
18578 0620   L1:LD      B,32            ; 32 Columns.
18580 3600   L2:LD      (HL),0          ; Store a Space character.
18582 23         INC     HL              ; On one.
18583 10FB       DJNZ    L2              ; Loop back (-5) to complete a column.
18585 3676       LD      (HL),118        ; Store a Newline character for end of
line.
18587 23         INC     HL              ; On one.
18588 0D         DEC     C               ; Decrement line counter.
18589 20F3       JR      NZ,L1           ; Jump back (-13) until all 24 lines
created.
18591 221040     LD      (16400),HL      ; HL now points to new VARS value so store
it.
18594 3680       LD      (HL),128        ; Store end marker for VARS area. VARS has
now been effectively CLEARed!
18596 23         INC     HL              ; On one.
18597 221440     LD      (16404),HL      ; Store E_LINE. End of BASIC. Nothing in
RAM above this value is SAVE'd or visible to BASIC.
18600 367F       LD      (HL),127        ; Set Cursor character in Edit area.
18602 23         INC     HL              ; On one.
18603 3676       LD      (HL),118        ; Edit area setup with only Cursor and
Newline.
18605 23         INC     HL              ; On one.
18606 221A40     LD      (16410),HL      ; Store new STKBOT
18609 221C40     LD      (16412),HL      ; Store new STKEND ; Routine has now done
its job. BASIC area stable and all game data safely stored above RAMTOP.
18612 D3FE       OUT     (254),A         ; Switch NMI generator on. Stack moved and
DFILE area setup so safe to enable interrupts again.
18614 3EFB       LD      A,251           ; Value for keyboard Row Q, W, E, R and T.
18616 DBFE       IN      A,(254)         ; Read from keyboard input port.
18618 CB47       BIT     0,A             ; Test for "Q" key being pressed.
18620 C8         RET     Z               ; Return to BASIC via address stored on new
stack, if so.
18621 C34945     JP      17737           ; Main Rock Crush game entry point. Note:
Dans Revenge uses JP 18012 instead.
```

End.

```
            The Steven McDonald "16K ZX81 High Res Range"
            ---------------------------------------------


      €€€€  €€€€  €€€€  €  €      €€€€  €€€€  €  €  €€€€  €  €
      €  €  €  €  €     € €       €     €  €  €  €  €     €  €
      €€€€  €  €  €     €€        €     €€€€  €  €  €€€€  €€€€
      € €   €  €  €     € €       €     € €   €  €     €  €  €
      €  €  €€€€  €€€€  €  €      €€€€  €  €  €€€€  €€€€  €  €

                         €€   €  €  €€€
                        €  €  €€ €  €  €
                        €€€€  €€€€  €  €
                        €  €  € €€  €  €
                        €  €  €  €  €€€

 €€€    €€   €  €  €€€€      €€€€  €€€€  €   € €€€€  €  €  €€€€  €€€€
 €  €  €  €  €€ €  €         €  €  €     €   € €     €€ €  €     €
 €  €  €€€€  €€€€  €€€€      €€€€  €€€    € €  €€€   €€€€  € €€  €€€
 €  €  €  €  € €€     €      € €   €      € €  €     € €€  €  €  €
 €€€   €  €  €  €  €€€€      €  €  €€€€    €   €€€€  €  €  €€€€  €€€€

∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞∞
```

High Res Range 1 - "Rock Crush" (C)1985 Steven McDonald, ZX81 16K
****

Loading Instructions:

Type in the following: LOAD "CRUSH"
For best results, place volume control at 75% of maximum. The program is recorded on both sides of the tape.

The Aim Of The Game:

You must guide Diamond Dan through the 10 caverns, whilst collecting all the diamonds, but beware of the rocks. If Dan stands under a rock for too long he will be crushed.

Note: When a life is lost, hold down any key. Press Newline to die!

Credits:

Programming: S. McDonald

Game Design: S. McDonald
Graphics: S. McDonald

High Res Range 2 - "Dan's Revenge" (C)1986 Steven McDonald, ZX81 16K
****

Loading Instructions:

Type in the following: LOAD "REVENGE"
For best results, place volume control at 75% of maximum. The program is recorded on both sides of the tape.

The Aim Of The Game:

You must guide Diamond Dan through the 10 caverns, whilst avoiding all the monsters along the way. Once you have collected all the diamonds a secret door appears allowing you to the next cave - if you can make it!

Credits:

Programming: S. McDonald
Game Design: S. McDonald & Scott Dolan
Graphics: S. McDonald & Scott Dolan

Rock Crush And Dans Revenge - The Definitive Edition Release: Sept 2018.

About The Games:

These two programs, "Rock Crush" and its follow-up "Dan's Revenge" were originally available for sale for the standard 16K ZX81 and featured a Spectrum quality high-res display resolution and fast 100% machine code action, all without extra hardware being needed.
In their time they received several rave reviews in magazines and coverage in Texas's WOAI 1200 Radio "Computer Line" program. The station also ran a competition where they were given out to listeners as prizes.
They were made available mail-order only and a lot of copies were posted abroad to the States and Canada (as well as here in the UK). The profits

made (over a few years) were quite reasonable for ZX81 software!
Looking back, however - they were written just that little bit too late
and the market had long shifted over to the superior ZX Spectrum.
Rock Crush dates from 1985 and Dans Revenge from 1986.
Both were hand crafted in 100% machine code and offered fast and smooth
performance.
In the modern computing world compared with 3D games such as Doom and Quake,
they do seem very simplistic and dated. However it still brings a tear to my
eye to see these games restored and working on a PC!
I would say that Rock Crush's best point is the screen design, which even
today, I like to see people puzzle over. I would have liked stronger end
sequences etc. but I was very happy with the finished program size - it had
to run well in 16K.
Unfortunately, with Dan's Revenge, I was more interested in my new optimising
technique for squeezing a 6K high-res screen design along with its monsters
into just 25 bytes per screen. I did plan to do 100 screens and memory space
was reserved for this.
I designed one screen and left the rest to Scott Dolan, who was going to
design and play-test the game.
As all the game engine was coded first and only the screen data had to
be entered, I left the designing of the levels up to Scott. He decided that 10
screens was plenty and that I should "patch" the code so that only 10 screens
had to be completed to finish the game. He figured 100 was too excessive.
With the benefit of hindsight, I have to agree! Anyway he did a good job with
a rather limited game engine!
When moving in Dan's Revenge - your timing must be exact - there is no
leeway! Tip: Try to move immediately after the monster has moved.

Eventually . . . the bottom finally fell out of the ZX81 market and the
hardware, software and users began to disappear.
One day, my own ZX81 finally gave up the ghost and was put in storage
with all my software and games with it - all to be forgotten about!
The ZX81 was gracefully laid to rest.

This mists of time have passed however. . . and they are now once again
being made available to ZX81 users.

I have decided therefore to make both these games available as FREEWARE.
If you like them - drop me an e-mail or whatever.
Did you have them the first time around?

I have had a lot of response from ZX81 enthusiasts from around the world
regarding these games.
It is good to know that they are still appreciated by people out there!

A short story on how they were rescued:

For this release, I was able to use a real ZX81 and the original Master
Duplicating Tapes. I used a ZXPand unit to save the programs to the PC.

The code is the original and no delays or modifications have been made
such as was made in the earlier internet releases for mainly the early
ZX81 emulators benefit. I hereby officially retire all previous releases!

Technical Information:

A lot of people used to wonder why both Rock Crush and Dans Revenge loaded
in five separate modules. The answer is that both games were written in
machine code on a 16K machine and with only that amount of memory, some
juggling is needed to fit an assembler, the data, source code and object
code into 16K whilst writing the thing!
My solution was to write a custom routine (which loads first). This
manages to set RAMTOP to a low value and to do it without a system reset
or 'NEW' which were always needed when you tried to do this on the ZX81.
Basically, the system stack and certain system variables are adjusted and
copied down. I was very happy with this routine. Each module then loads
and runs a routine to dump it above RAMTOP and then loads the next one.
The fifth module is the game code which then begins executing at last -
with everything in place!
All of this works fine under the emulator.
If you want to nose around, Pressing "S" as each module loads (you need
to be very quick), will allow you to break in and examine the code.
The original had piracy protection code which crashed the machine if you
attempted to stop execution. This has since been removed!

For those interested, the hi-res screen display is generated by an interrupt
driven display routine which replaces the normal ZX81 screen generation
code. This routine can only generate pseudo-high-res with a limited number
of bit-patterns available. With perseverance however, quite passable graphics
can be designed. Basically, there is true high-res vertically, but not
horizontally, due to the ingenious ZX81 design economies built into the ULA
chip.

For me that is one of the most fascinating features of the ZX81, the fact
that it generates its own video signal, using nothing more than the Z80,
some extremely clever interrupt driven routines and a ULA chip to glue
everything together. Not many computers did that - but it did create the
possibility for re-writing the routines and messing about with the hardware
in general to create some very weird effects!
This is something you could not really do on a ZX Spectrum as its ULA
tended to take much more control of the machine. Whereas on the ZX81, under
tightly controlled code, the programmer could fool the ULA into doing some
strange things that it was not designed to do!

See the included file HIGHRES.TXT for a more detailed explanation of the
software only ZX81 high-res system.

Contacting the author:

If you have any questions or comments, regarding these games or any other general matters, feel free to contact the author, Steven McDonald who can be reached via e-mail at:

support@rock-crush.com

Version History:

Rock Crush and Dans Revenge V1.0 - 1985 and 1986 (Original Cassette release)
----

* Hi-res core routine developed and optimised first and then Z80 'I'
  register value decided upon (this determines the bit-patterns available).
* Rock Crush Graphics and game design worked out.
* Rock Crush game engine coded and screens designed.
* Rock Crush opening screen designed and stored in display buffer.
* Rock Crush duplicating master created.
* Dans Revenge graphics worked on and game design worked out.
* Dans Revenge game engine coded and screens designed.
* Dans Revenge opening screen designed and stored in display buffer.
* Dans Revenge duplicating master created.
* Games tested extensively on standard 16K ZX81's and all bugs removed.
* Games sold initially as mail-order and sent out around the World.
* Later master had piracy protection code removed.

Rock Crush and Dans Revenge V2.0 - 18th June, 1996 (Re-mastered version)
----

* 16K ZX81 restored to operating condition and original cassette
  duplicating masters located after 10 years!
* Custom PC ZX81 Encoder and Decoder successfully developed and tested.
* Game files transferred from the ZX81 to the PC and formatted for use
  with Xtender.
* Extensive testing to check compatibility with Xtender - no problems!
* Code added to stop loading each module when "S" key is held down.
* Game Covers Photo scanned-in and edited.
* Batch file written to match speed of emulator to original.
* Documentation file written.
* Zip file uploaded to the Internet and games released as Freeware.

Rock Crush and Dans Revenge V2.1 - 31st July, 1996: (Updated Re-master)
----

* HRESGO.BAT - changed delay from 115 to 127 to better match speeds.
* Code added to Dans Revenge to detect whether the game is being run on a
  real ZX81 or under the emulator and to adjust the game speed accordingly.

```
 * The separate game modules (except final code modules) re-mastered with
   collapsed display files to save space and provide faster loading times
   (as in the original cassette release).
 * Documentation file corrected and updated.
 * New version uploaded to my new ZX81 web site and released as Freeware.

Rock Crush and Dans Revenge V3.0 - 3rd July, 1997: (Win95 and new timings)
--------------------------------------------------------------------------

 * BASIC Code added in point 2 above of the V2.1 release - totally removed!
 * Empty delay loop in both games - patched to wait 4 TV frames instead.
 * Updated title screen on both games to display V3.0
 * Tested exclusively under Windows95 - much better game performance.
 * Documentation file corrected and updated.
 * Added HIGHRES.TXT - a detailed explanation of the ZX81 High Res system.
 * Changed name of ZIP archive from 'ZX81HRES.ZIP' to 'ZX81HR95.ZIP' for V3.0!
 * New version uploaded to my new ZX81 web site and released as Freeware.

Rock Crush And Dans Revenge - The Definitive Edition Release: Sept 2018.
------------------------------------------------------------------------

* All previous releases officially retired.
* A new fresh master loaded in from the original Duplicating Master Tapes.
* Nothing modified regarding original game code/play feel/timings etc.
* A new routine added and documented here to allow a one-part load.
* This new routine manages moving memory about but does not interfere
  with the operation of the game itself and indeed goes to great lengths
  to leave the machine in the same state as the original 5-part load.
* Documentation updated.
* Remastered Inlay Cards - same as original only tidied up even more!
* A new batch of cassette tapes run of and available for sale.
* New version uploaded to my new ZX81 web site and released as Freeware.

--------------------------------------------------------------------------

License:
--------

Both these programs have been released as Freeware. However the author retains
the copyright in these works. Feel free to duplicate and pass on these
programs as you wish. However, I ask that this archive is passed on intact and
that all files including this README.TXT are included as originally packaged.

Thank You.

Steven McDonald.


Rock Crush And Dans Revenge - The Definitive Edition Release: Sept 2018.
------------------------------------------------------------------------
```