FIRST STAR SOFTWARE INC.

BOULDER DASH

BOULDER DASH®

A Game for the Atari 2600/7800

Programmed by Andrew Davie and Thomas Jentzsch

Co-Published by AtariAge and First Star Software, Inc.

Contents

Well, it's a (very) long story...

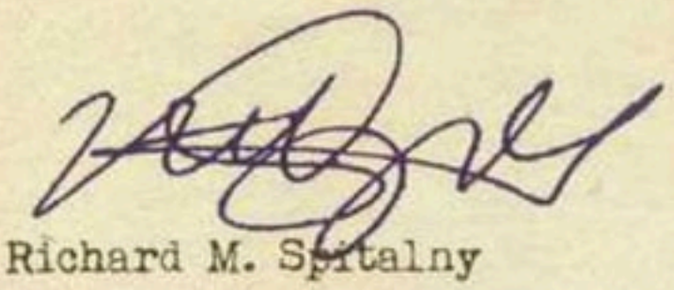
Boulder Dash® Vol. 1 for the Atari 2600/7800 was many years in the making. It's only being released in 2012 after several years of 'false starts', now that the stars seem to have all (finally) aligned, and just as First Star Software, Inc. kicks off its 30th year anniversary! Part of a very limited edition, it's a small miracle that the cartridge you now own was ever manufactured. Think about it: Boulder Dash was first released in 1984, twenty-eight years ago - some 7 years after the Atari 2600 first went on sale.

So, how did this come to pass? The vast majority of the credit needs to go to Andrew Davie and Thomas Jentzsch who both, despite the incredible limitations of the hardware, were determined to program their way around those restrictions to prove (to themselves and others) that they could squeeze the hardware harder than it had been squeezed before. They wanted to create an amazingly 'true' port of Boulder Dash for this ancient and much revered platform; and, they wanted to share their code and how they did that with other 'homebrewers' and members of the Atari community while always being respectful of our Intellectual Property rights.

Of course, in order to attain their goals Thomas and Andrew needed to fit this project into their 'day to day' lives. This lead to various degrees of focus on and off over the past few years, while I discussed with Andrew First Star Software's objection to releasing source code that might further add to the materials available on the Internet used by others who were not as forthright when it came to respecting our IP. Sporadic emails were thus exchanged about whether this project would simply be an intellectual 'proof of concept' with videos and/or a single cave made available to the Atari community; or, perhaps it could be something more 'commercial', say the release of a limited number of cartridges to be made available for sale to collectors.

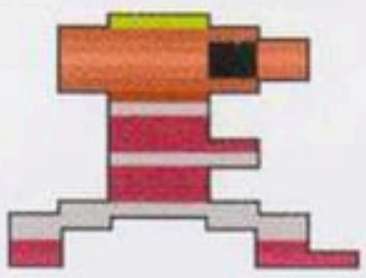
The end to the story is now clear but not without further thanks and credit to Albert Yarusso and AtariAge who, working with Andrew, Thomas, André Bolfing (winning concept artist) and Nathan Strum (layout design artist) have come up with a box, label and manual that I truly believe any collector will be thrilled to own! Thanks everyone.

ENJOY!!



Richard M. Spitalny
President and Co-Founder
First Star Software, Inc.

Game Objective



The objective of Boulder Dash is to guide Rockford™ through many challenging caves and collect as many diamonds as you can in the shortest time possible. Once the required number of diamonds have been collected, the door to the mysterious escape tunnel is revealed and you can exit to the next cave. Strategy and planning will help you master the "physics" of Boulder Dash. Boulders drop predictably enough, but you also have to block growing Amoebas, transform Butterflies, outmaneuver Fireflies, and overcome numerous other obstacles.

Game Controls

- For a one-player game, plug a joystick controller into the left joystick port. For a two-player game, you can either plug a second joystick controller into the right joystick port, or take turns using a single controller.

- When viewing the title screen, press the joystick button to go to the selection screen (see: **Selection Screen** for selecting starting cave and level of difficulty).

- During gameplay, move the joystick in the direction you wish to move Rockford.

- To push, grab or dig without moving Rockford, hold the joystick button down while moving the joystick in the desired direction.

- To look-around the cave, without moving Rockford, hold the joystick button down. After a moment, the screen background color will change to grey, and you can then use the joystick to look-around. Release the joystick button to resume playing. While you are looking around the game is still playing at half speed, so don't get killed or run out of time!

Console Switches

PAL/NTSC

Set the RIGHT DIFFICULTY switch to B to play Boulder Dash on NTSC consoles, and to A to play on PAL consoles. For PAL set the LEFT DIFFICULTY switch to B for standard 50Hz display, and to A for 60Hz display. The latter will result in a taller display but may not be supported by all PAL TVs.

Changes to the DIFFICULTY switches will not take effect until Rockford re-enters the current cave (through loss of life), or enters a new cave (through cave completion or end of game).

Pause

To pause and resume the game, toggle the TV TYPE switch (Atari 2600) or press the PAUSE button (Atari 7800). The screen background turns red while the game is paused.

Trapped

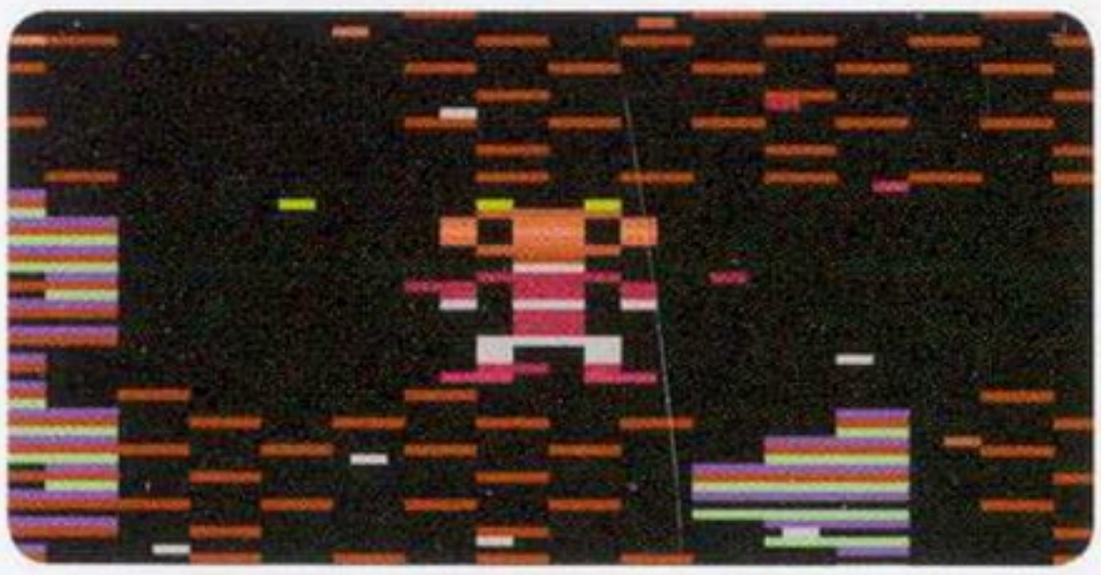
In order to start the current cave again when hopelessly trapped, press either the RESET or the SELECT switch. You will lose one life when you do this!

Restart

Press the SELECT and RESET switches simultaneously to end the current game and return to the Selection Screen.

Lives

Each player starts with 3 lives. Bonus lives are awarded after every 500 points. The cave will "sparkle" when you are awarded a bonus life. A bonus life is also earned by completing a Playable Intermission.



Selection Screen

Selection of game options is performed using the left joystick. Move the joystick up or down to highlight the option you would like to change. Move the joystick left or right to change the currently highlighted option.

After all options have been configured, press the joystick button to start the game.

Selecting the Starting Cave

There are 16 caves labelled A through P, each comprised of a large playing area. Only A, E, I or M can be selected as the starting cave. To select a different cave, move the joystick left or right when CAVE is highlighted.

Selecting the Difficulty Level

Each cave has 5 difficulty levels. To select a different difficulty level move the joystick left or right when LEVEL is highlighted. The greater the difficulty, the less time you have and the more diamonds you have to collect before you can exit the cave. The placement of diamonds and boulders changes with the difficulty level. On difficulty levels 4 and 5, you must start with cave A.

One or Two-Player games

You can select a one or two-player game by highlighting PLYRS and using left or right on the joystick to change the number of players.

Two-player games alternate turns between players at loss of life, as long as the other player has lives left. After loss of life, and a press of the joystick

button to continue, the other player's turn commences and the score line will display either P1 (for Player 1) or P2 (for Player 2), indicating which player is playing.

Joystick Selection

During a two-player game, the players may share a joystick, or each use their own joystick. To select the one-joystick or two-joystick option, first select a two-player game, highlight the JOYST line and press left or right on the joystick to select the desired number of joysticks.

Caves

This section describes what you can expect in each cave. If you want to explore Boulder Dash on your own, skip over this section.

✓ **A. Intro:** Pick up diamonds and exit before time is up.

B. Rooms: Pick up diamonds, but you must move boulders to get all diamonds.

C. Maze: Pick up diamonds. You must get every diamond to exit.

D. Butterflies: Drop boulders on Butterflies to create diamonds.

✓ **E. Guards:** The diamonds are there for the grabbing, but they are guarded by the deadly Fireflies

F. Firefly Dens: Each Firefly is guarding a diamond.

G. Amoeba: Surround the Amoeba with boulders, so it can't grow anymore. Pick up diamonds that are created when it suffocates.

H. Enchanted Wall: Activate the Enchanted Wall and create as many diamonds as you can.

✓ **I. Greed:** You have to get a lot of diamonds here; lucky there are so many.

J. Tracks: Get the diamonds, avoid the Fireflies.

K. Crowd: You must move a lot of boulders around in some tight spaces.

L. Walls: You must blast through walls to get at some of the diamonds. Drop a boulder on a Firefly at the right time and place to do this.

✓ **M. Apocalypse:** Bring the Butterflies and Amoeba together and watch the diamonds fly.

N. Zigzag: Magically transform the Butterflies into diamonds, but don't waste any boulders and watch out for the Fireflies.

O. Funnel: There is an Enchanted Wall at the bottom of the rock tunnel.

P. Enchanted Boxes: The top of each square room is an Enchanted Wall, but you'll have to blast your way inside.

(✓ = selectable from Selection Screen)

Scoring

You score points for collecting diamonds, and for remaining time when you complete a cave. The number of points per diamond changes - the point value is

determined by the cave, difficulty level and bonus status.

You also score 1 to 5 bonus points (depending on difficulty level) for each second of time remaining when you successfully exit a cave.

During gameplay, the scoring bar located on the top of the screen displays the following information:

Player (top left)	P1
Lives	3
Cave and Difficulty Level	A1
Diamonds Required	012
Extra Diamonds (after required ones have all been collected)	005
Time Remaining	149
Score	000644

High Scores

Your high scores are saved permanently when a SaveKey or AtariVox is plugged into the right-hand joystick port. The previous high score is displayed in red after a game is completed.

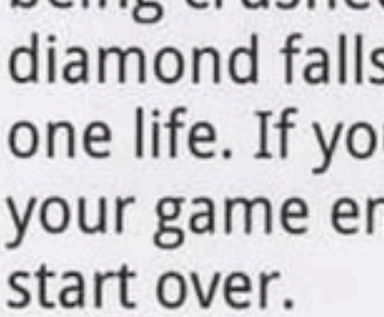
Playable Intermissions

There are four interactive puzzles you are entitled to play after completing caves D, H, L or P. There is no penalty for not playing or losing the intermissions, but if you complete an intermission successfully, you receive a bonus life.

Minor changes to intermission dimensions have been made to improve gameplay on the restricted Atari 2600 screen.

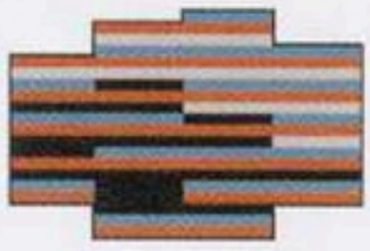
Game Elements

Rockford



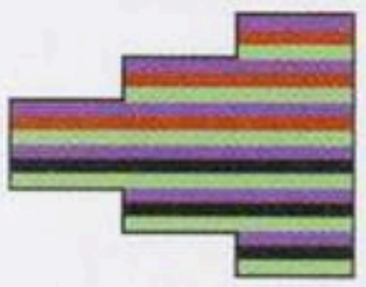
This little fellow is the star of the game! Rockford has the power to dig through the earth and collect the diamonds found along the way. He can push a single boulder horizontally if there is nothing to block its path. Rockford can stand directly under a boulder without being crushed, but if a boulder or diamond falls on him, you will lose one life. If you lose all of your lives, your game ends and you'll have to start over.

Boulders



The boulders will fall whenever gravity dictates. They will fall straight down if unsupported, or they will topple off underlying objects if there is nothing to block their way. Learning how the boulders fall is the best way to learn the "physics" of Boulder Dash.

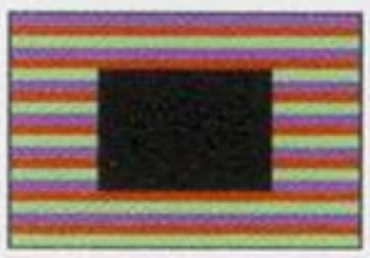
Diamonds



You must collect the required number of shimmering diamonds in order to exit one cave and advance to the next. There are many ways to create diamonds. For example:

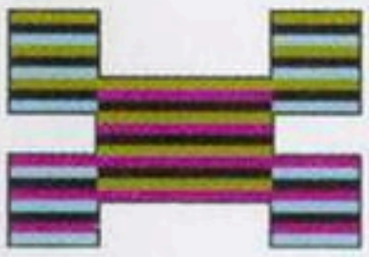
transforming Butterflies, suffocating Amoebas, or dropping boulders through an Enchanted Wall will create additional diamonds. When the proper number of diamonds have been collected, a flash will let you know that the Escape Door is revealed and open.

Fireflies



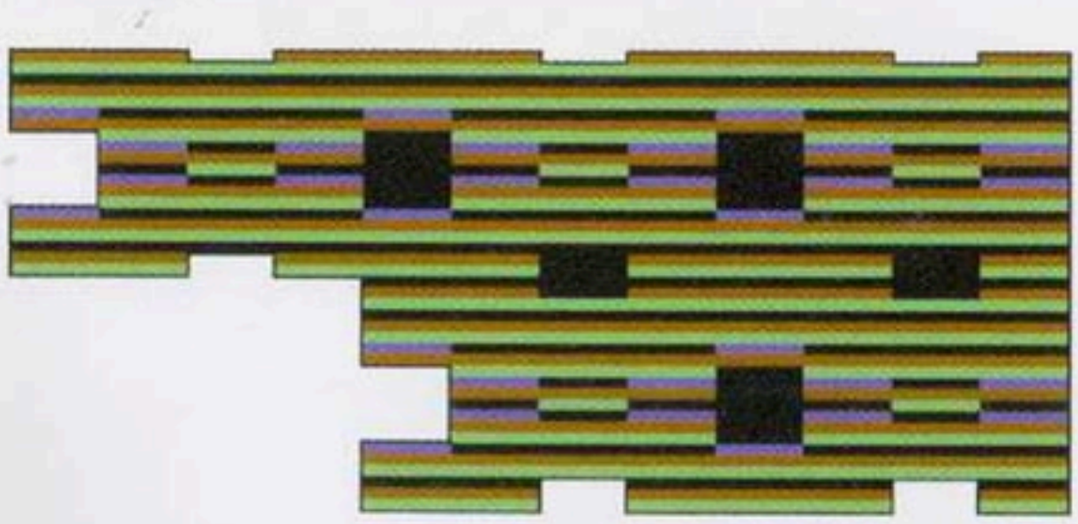
Beware the deadly Fireflies; they will explode on contact with Rockford. Helpful hint: their behavior is predictable as they only move along the edges of the exposed areas. Try turning the tables by dropping boulders on them, which causes them to blow up. This comes in handy when you want to demolish a wall in order to get some diamonds.

Butterflies



The colorful Butterflies behave much like Fireflies. However, they fly in the opposite direction of the Fireflies and they turn into diamonds when they explode.

Amoeba

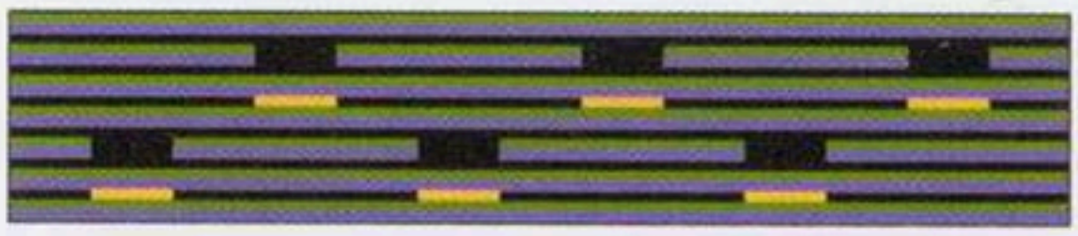


The Amoeba is a green blob that bubbles and grows through earth and air. Rockford can touch it without harm. Fireflies and Butterflies will explode on contact with the Amoeba. When the Amoeba is surrounded and cannot grow any further, it suffocates and turns into diamonds. However, if the Amoeba grows too large (about 200 squares big) it will die and turn into boulders.

Enchanted Wall

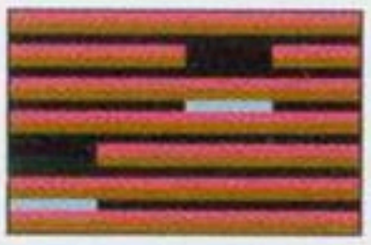
The Enchanted Wall looks like any other wall; however when hit by a falling boulder it begins to vibrate for a limited time. During this period, any boulders that drop through it are magically turned into diamonds, but only if there is empty space below the wall. Be careful, though, because the Enchanted Wall also turns diamonds into boulders! Once the enchantment phase is complete, it cannot be reactivated.

Titanium Wall



The exterior wall is an indestructible boundary to play action. Rockford can only exit to the next cave through the Escape Door.

Escape Door



Initially the Escape Door looks like a portion of the Titanium Wall. After Rockford has collected the required amount of diamonds, it is activated and begins to flash. At this point the

mysterious Escape Door is revealed and you may exit, prior to time running out.

Time

Each cave is timed. When time is running out, a series of warning sounds will occur, and the timer will start to flash. When there is no remaining time, and you have failed to complete the cave, you lose a life.

Strategy/Tips:

Rockford

Rockford can affect an object that is next to him without moving into it. Keep the joystick button depressed and move the joystick in the direction of the object or earth you wish to affect. Rockford will not move, but the object will react. Use the push, grab and dig moves to save time and reduce the risk of dying.

Boulders

You will often find yourself digging or moving downwards only to find that a boulder has been toppled by your movement and is about to land on you. A boulder (or diamond) starting to fall will make a sound. So listen carefully - if you hear a boulder falling unexpectedly, you'd better look out!

The only way to avoid losing a life in these situations is to move quickly to the right or left, out of the boulder's way. Always watch for an escape route in case boulders should start falling on you.

Boulders fall at the same speed as Rockford can run, so they will never catch up with you unless you hesitate or stop.

Escape Door

When Rockford has collected the required amount of diamonds and the Escape Door is revealed, you should decide whether to go for bonus diamonds (at a higher value) or to exit for the time bonus points.

Make sure you know the location of the Escape Door and that you can get to it before time runs out.

Additional Hints

- Use the look-around function with caution. Try to memorize the caves - you'll be able to successfully complete them in less time.
- When surrounding an Amoeba, make sure you can get to the diamonds when it suffocates. One option is to use Rockford to serve as a barrier to the Amoeba.
- For Enchanted Walls, create space below and move and collect boulders together before you activate the wall.
- Try building paths to trap Fireflies and Butterflies.
- When facing a lot of enemies, create enough space to be able to evade them.

The Journey

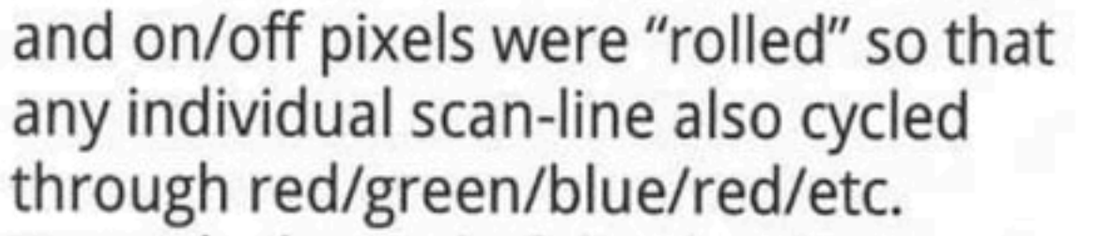
It has been a long, hard journey building the completed product you see before you.

The Beginning

It all started around December 2002 when Thomas, Andrew and others were wondering if Boulder Dash was possible on the Atari 2600.

They had, around that time, been experimenting with techniques to display colored bitmap images, using multi-color 6-sprite routines. These experiments resulted in the Interleaved ChronoColour (ICC) demos. The colored images were displayed by having different colors on successive scan-lines (red, then green, then blue, then red.... etc.) and considering a virtual pixel as a combination of on/off pixels covering those three lines. The scan-line colors and on/off pixels were "rolled" so that any individual scan-line also cycled through red/green/blue/red/etc. Towards the end of the development of these interesting color display techniques, when Thomas and Andrew were in friendly competition to find the best way to produce colored graphics, they started discussing ways to display graphics for a Boulder Dash game.

Thomas presented a simple mock-up display using multi-color interleaved sprite graphics in December 2002.

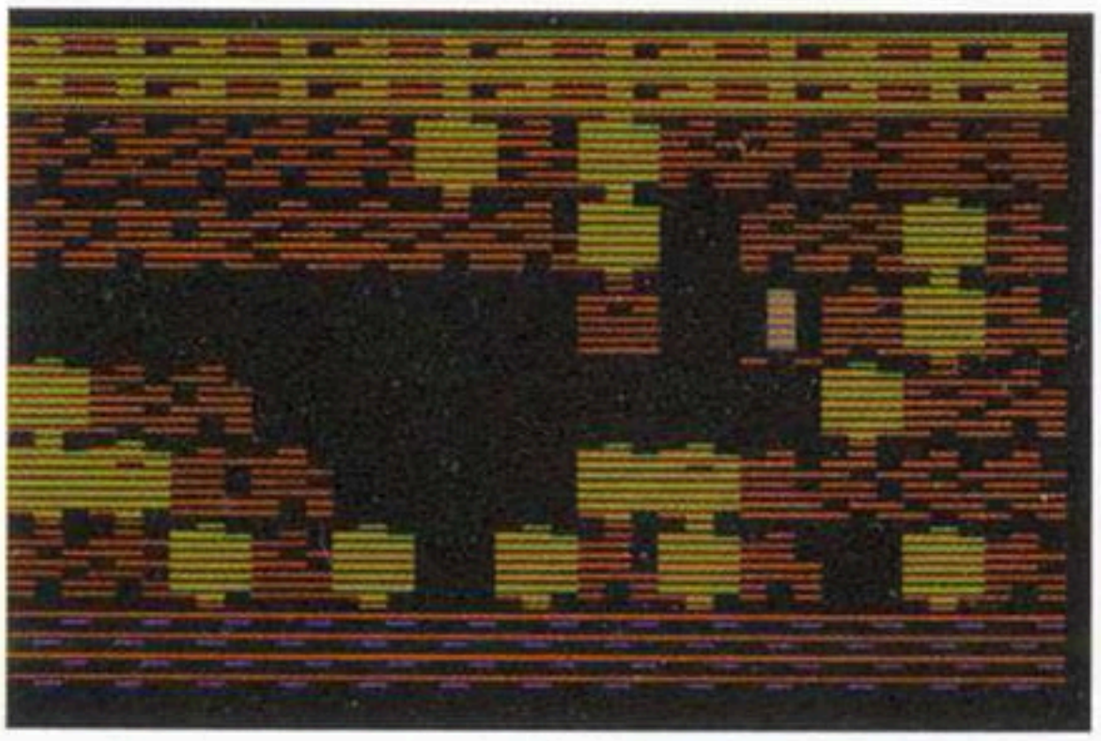


Thomas's mock-up demo using sprites, December 2002

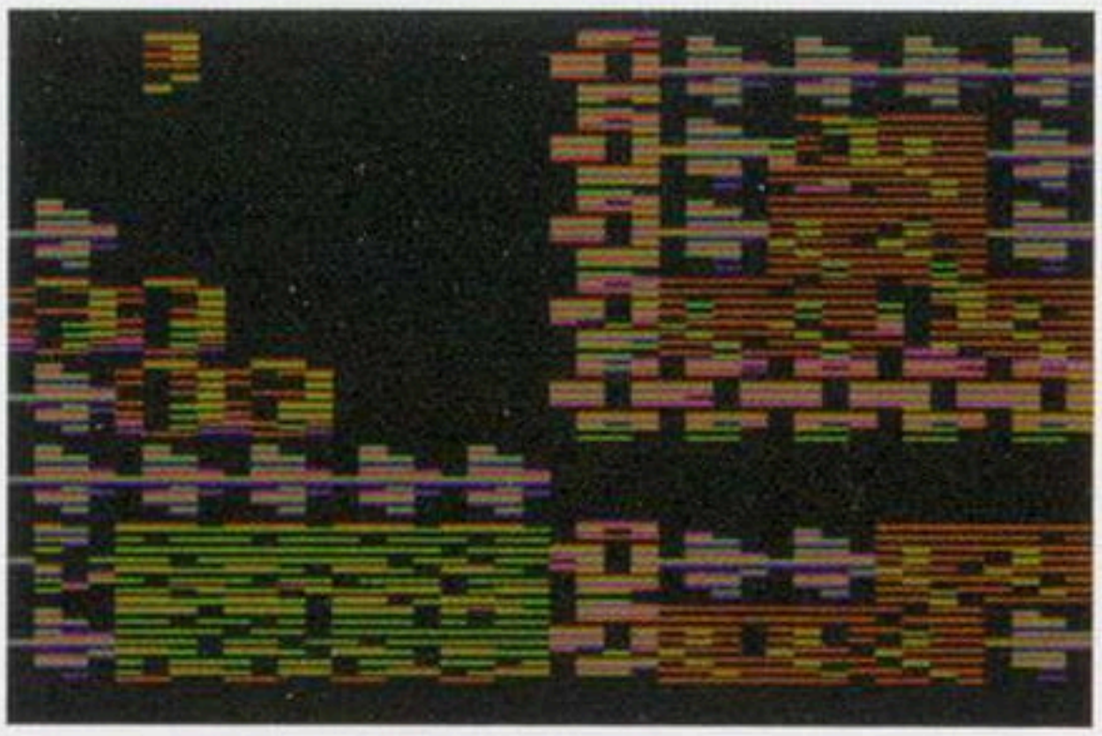
It looked quite nice, but at that time I was still more into staying within the limits of the mid-80's. And back then, the amount of RAM that Boulder Dash needs was too expensive to add it to a cart.
- Thomas

Andrew continued his line of experimentation with full-screen playfield graphics using the ICC technique. The limited horizontal playfield resolution (40 pixels) would mean that the objects in the game would need to be drawn using just a few pixels each. These early mock-ups were exploring how those objects would look. With 3 pixels per object, 13 objects could be drawn on a line. It was hard to see how recognizable objects could be drawn with just 3 pixels. With 4 pixels per object, just 10 objects could be drawn on a line, but objects could (just) be recognized.

These demos all flickered, relying on rapidly changing pixels and colors to achieve the multicolored graphics.



13-character mock-up. Objects are unrecognizable. April 2003 - Andrew



10-character wide mock-up. Surprisingly similar to the completed game! April 2003 - Andrew

The Display Engine

Mock-ups are one thing, but how on earth could this sort of graphic display actually be implemented along with all of the processing and game logic required for Boulder Dash? It's a seemingly impossible task for a 1.19MHz processor which has to spend most of its life feeding graphics to the TV.

Clearly, representing a 40 x 20 character-based scrolling display at ~20Hz is not going to happen on a 1.19MHz 6507 with just 128 bytes of RAM. - Andrew

The first concession to reality was to relax the RAM limitation. The Atari 2600 has just 128 bytes of RAM, but various bank-switching schemes had previously added extra RAM, so it was not unreasonable for a Boulder Dash cartridge to use extra RAM.

Both Thomas and Andrew were fortunate to have the wonderful Krokodile Cart (KK) – designed by Armin Vogl. This is an Atari 2600 cartridge which emulates other cartridges. In short, you can download any game to it, using any bank-switching scheme, and the KK is able to emulate the bank-switching scheme so that game can play on the KK. Great stuff! The KK is a fairly capable bit of hardware, wonderful for development, and to allow it to emulate all of the various bank-switching schemes it has a hefty amount of RAM on-board.

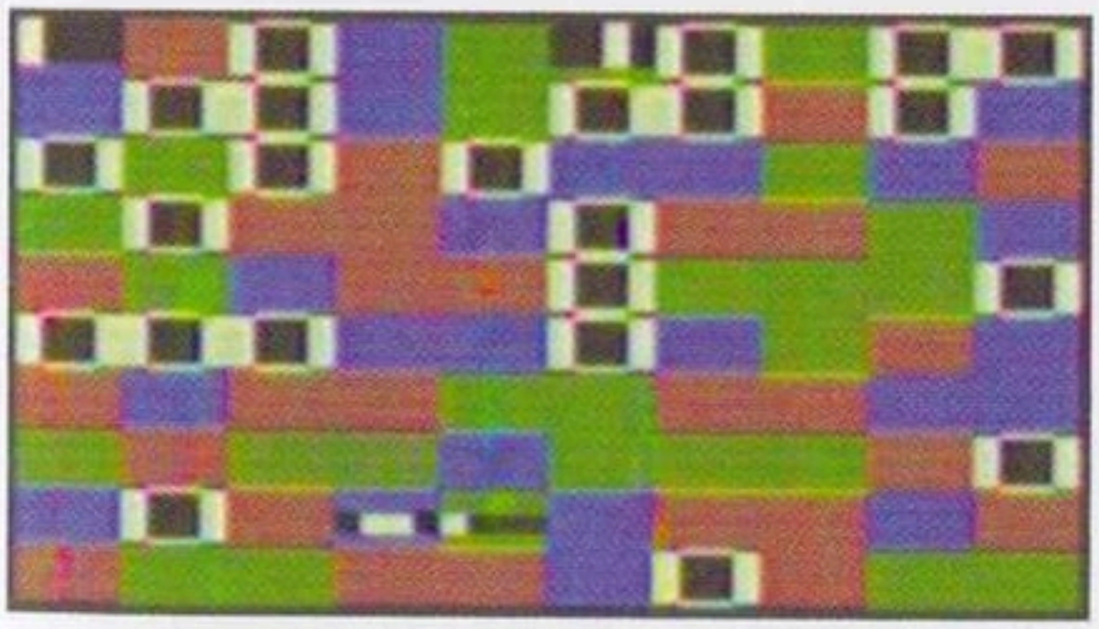
After implementing the logic for all supported bank-switching schemes, Armin found himself with a little left-over space and he designed (in consultation with Andrew) a new bank-switching scheme to assist with some of the early multi-color graphics demos that Andrew was working on (e.g., color dancing baby). This new scheme was known as "3E", which is the address to which writes are made to do the actual RAM bank-switching. This new scheme is limited by available code space Armin had in the KK ROM (very little), and so there were some limitations related to writing data across pages, and the organization of switch-able banks was very restrictive. But it provided RAM... lots of it... and so it was attractive!

In hindsight it's a rather ugly and difficult to use bank-switching scheme, and we definitely wouldn't use it if we were developing the game from scratch.

One of the first design-decisions was to remove the rolling colors from the

display. This meant that the colors on the scan-lines were not 'rolled', and only a single 'frame' was used to draw objects, instead of three frames with alternating red/blue/green scan-lines. This saved significant memory, and simplified the draw systems. The resultant screens were not as high-resolution as the original system, but did not suffer from flicker. None whatsoever!

Andrew spent much of 2004 working on the underlying graphics engine. This engine does in software what most later consoles did in hardware; provided a character-mapped display. The 'characters' in the engine are the boulders, dirt, diamonds, etc. They are 4 playfield pixels wide (a nybble each) and 21 scan-lines deep. However, due to the triplet nature of the ICC-type display, the vertical resolution is ambiguous; 21 scan-lines but only 7 'RGB' pixels. The engine performs a massive amount of byte-shifting to draw the 40x20 scrollable playfield on which the Boulder Dash game is played. Early versions of the engine proved the feasibility of a character-graphics emulator but were so slow as to be effectively unusable.



Prototype character graphics scrolling playfield, December 2004

To draw a character on the screen you have to move at least 21 bytes to the screen. With 80 characters on the screen (8x10), that means a lot of bank-switching and byte masking. And soon you find that to update the entire screen (which you have to do when, for example, the game scrolls)... you are looking at multiple seconds per frame. The first solution to the scroll problem was to have a static screen which then halted the game when you got near the edge, redraw the whole screen, and then let play continue. It was ugly but at that time the only viable solution. All along, this game has been riding on the edge of the capabilities of the machine and the software.

One of the real problems for efficient screen draw is the very odd playfield format. The bytes are inconsistent, and the bits in the bytes go every-which way. It means that when you are drawing a character in one position on the screen, different shape data needs to be used than if it is drawn in another position on the screen. This, in turn, doubles the space you need for character shape definitions which in turn fills up the bank ROM space which in turn means you have less space for character definitions so you have to have less characters or do more bank-switching.

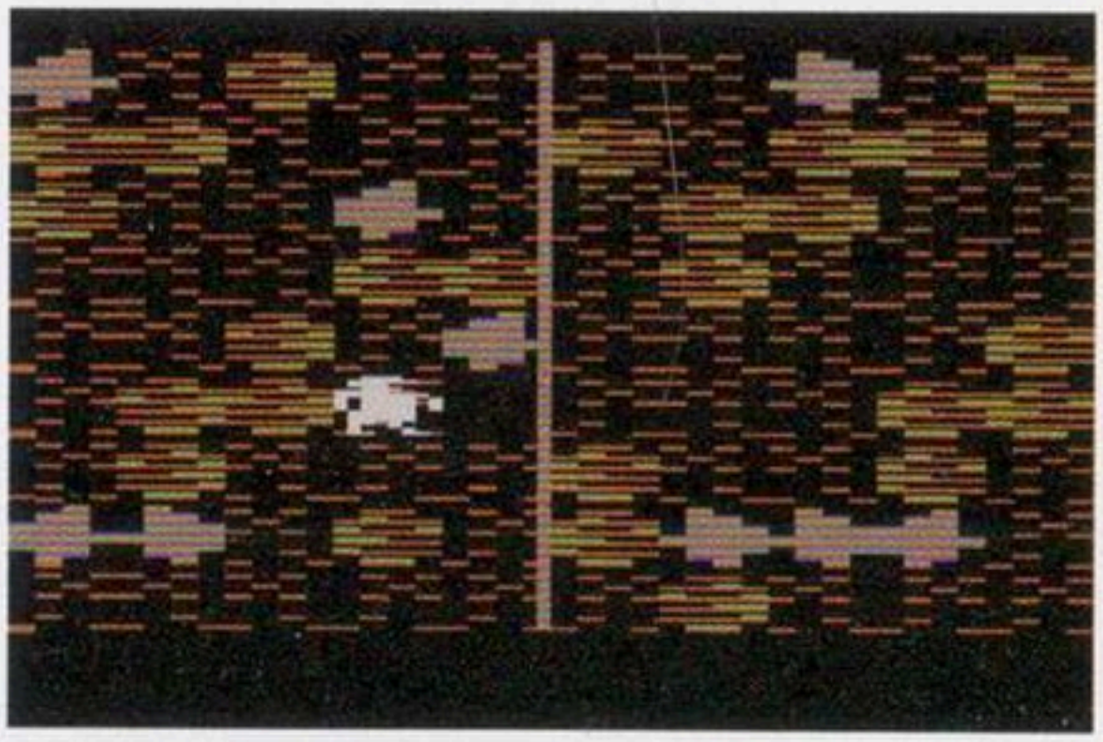
A major leap forward came with the idea of a 'differential screen draw'. The reasoning behind this is that most of the screen doesn't actually change when you scroll. Lots of dirt and a few boulders, and a few other items. But much of the screen is

the same -- even though your brain doesn't really notice this. The engine takes advantage of this and only draws those characters that changed. This reduces the draw requirements from 80 characters per scroll by a factor of ten. The system for the differential draw wasn't exactly simple, and this required extensive rewrites of the display system -- but it was well worth it. Now we had something which showed promise!

The Collaboration

Around this time I did two things that in hindsight were exactly the right thing to do. Firstly, I contacted Thomas and invited him to participate in the programming. And secondly, I contacted First Star Software (FSS) to ask for their permission to work on the game.

I was very impressed, offered my help and so our collaboration began.



Split-screen scrolling demo, May 2005 - Andrew

Thomas joined the project in May 2005, after Andrew posted a video on AtariAge showing his work to date. Thomas and Andrew worked very intensely for the following two months. Thomas first concentrated on peep-hole optimizations and was able to accelerate the drawing to the screen buffer quite nicely. He also rewrote a new main kernel and added the score display, and wrote the generator which creates the caves using the board data from the original games. In two short months, Boulder Dash went from a proof of concept demo to a playable game!

At the beginning I had real problems to understand what Andrew had been coding, even though he very elaborately described it to me (again and again).

Thomas extended the ICC color-triplet concept and designed a new color system which allowed each cave to have a unique color scheme. This made a big difference to the look of the game. The system involved switching from always mixing pure red, green and blue to defining three dedicated colors for each cave. The 1st color being simply the dirt color, the 2nd one is a mix color, which, together with the dirt color, mixes to the main steel wall and boulder color, and the 3rd color is always a bright color, which is complementary to the mix of the 1st and 2nd color. When all three colors are mixed together, the result is close to white.

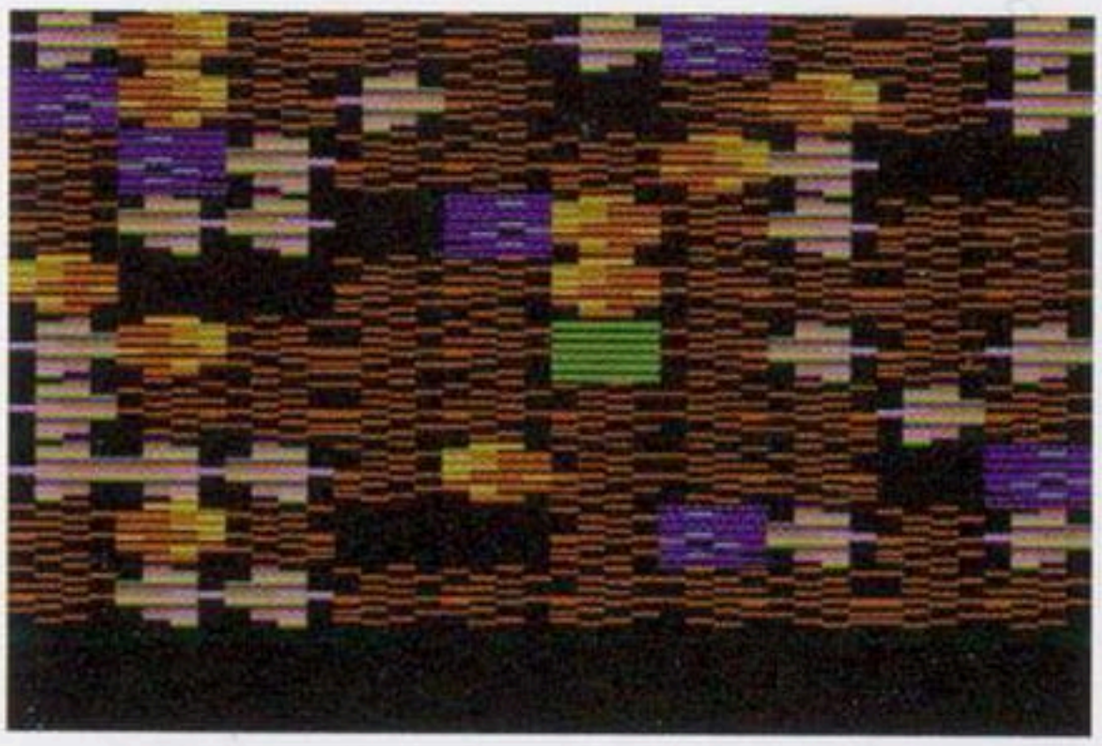
This gives the cave nice, unique, and contrasting colors. Other objects (e.g. the diamonds) use the remaining color combinations.

While I was very pleased with the results, it later turned out that Andrew didn't always agree. So the final color mix caused a lot of discussion between two pretty stubborn individuals. This lasted until the very end of the project. It was sometimes pretty exhausting, but in the end this and other discussions only made the whole thing better.

LOL @ 'lot of discussion'

Thomas and Andrew developed draw algorithms which were specifically tailored to the odd screen layout (for example, draw two characters at once, where possible, because the same shape is side-by-side in the same byte on-screen). There are quite a few of them... and they're ALL cycle counted by hand; it's known to the very cycle how long most of the draw algorithms take to do their stuff. Boulder Dash is possibly the most cycle-counted game on the machine... ever!

The real innovation which allowed the whole thing to work; this game operates by dividing all tasks into very small processing sub-chunks and running these sub-chunks only when there is sufficient processing time available to do them. So we need to know in advance just how long any chunk will take to do its stuff. All code is cycle counted. And what this all means is that when the screen is being drawn -- a common thing for all '2600 games -- there are certain parts of the display where you have a few cycles spare (for example, during the vertical blank when the TV beam is retracing back to the top of the screen). There's a few places where you can "do stuff". So the system uses a countdown timer which is initialized to indicate exactly how many cycles are available in each of these places where stuff can be done -- and then keeps calling the processing sub-chunks until there's no more time to fit a chunk in. Then it waits until the next place where stuff can be done. That's a way-simplified overview, of course, but in essence this game is a multitasking time-slice system built on top of a normal '2600 display kernel.

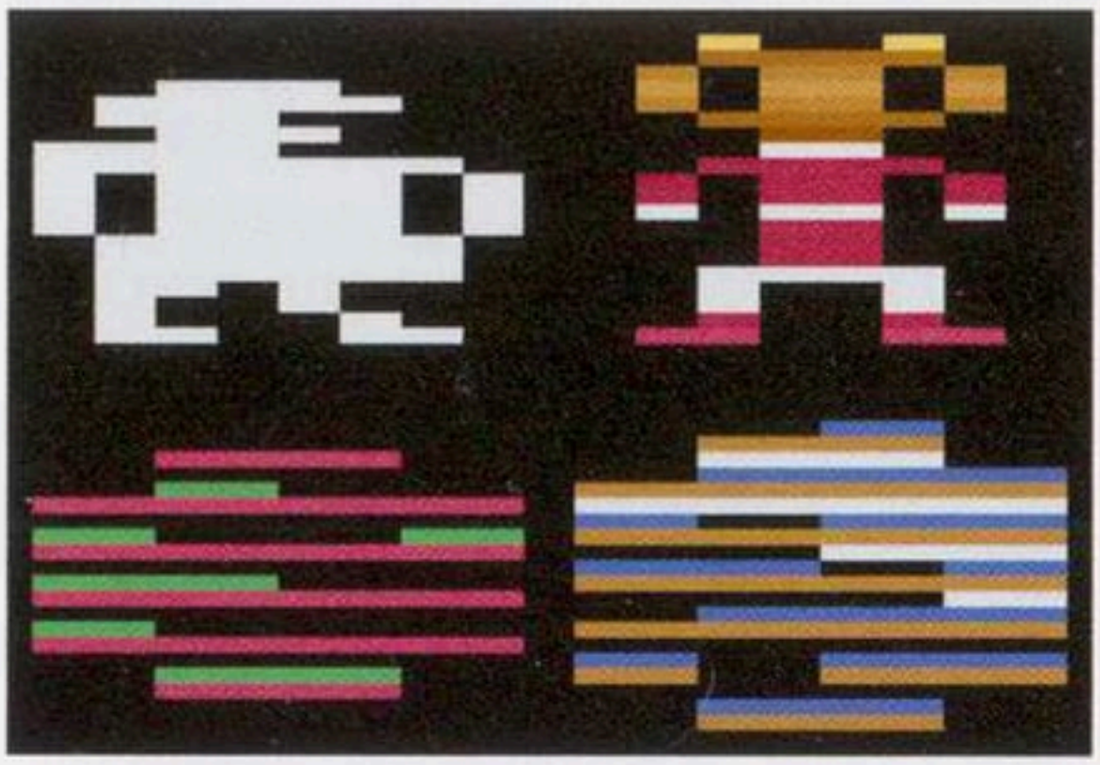


Early version of the character-graphics display, January 2005 - Andrew

Visually not very impressive, but the main structure is already there.

I'm great at systems and logic, but my graphics are mechanical and programmer-esque.

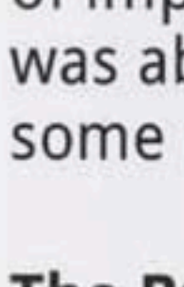
In mid 2005 Thomas redid the graphics, which up to that point were more or less just a stub to test that Andrew's system was working.



Before and after: Thomas's graphics rework, 2005

A scrolling algorithm similar to the one Thomas used in Thrust was added. And with Andrew doing a lot of improvements on his side too, it was about time to show the results to some people.

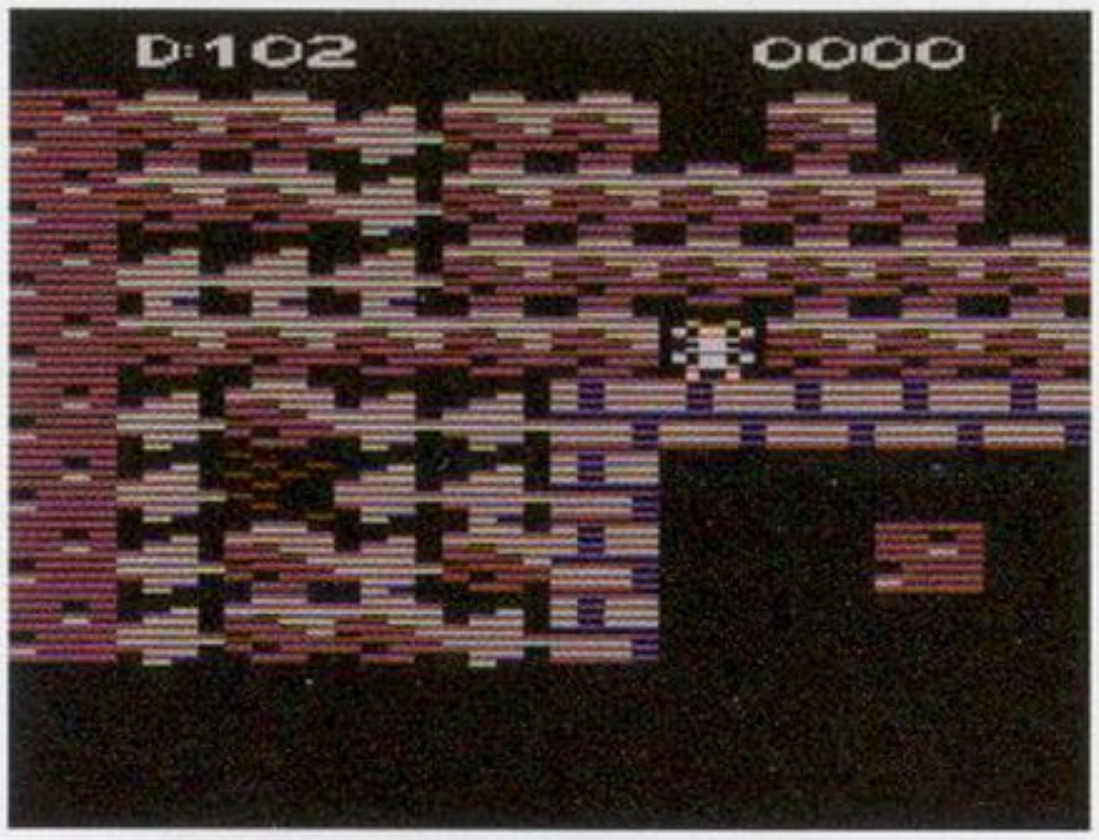
The Breaks



The project was abandoned by me at least a half-dozen times. It's always been a part-time thing, and sometimes it just takes over your life so much that you just have to let it go.

Richard from First Star Software sent back some very positive feedback and Andrew decided to show the game at a US show.

It must have been the wrong audience, since unfortunately here the lack of positive feedback was pretty disappointing, especially for Andrew. This caused the first major break in development. Andrew declared the project to be dead, but Thomas was hoping for the best.



First public demo of Boulder Dash, May 2005

This break lasted until late 2007 when Andrew became interested again and they restarted intense coding for another 3 months until the main gameplay was almost done. They often ran out of ROM space, but Thomas was able to use his experience from his 1k mini-games to find extra room.

Then break number two followed, and Boulder Dash was declared dead for the 2nd time. Again, Thomas was not prepared to concede that the project had ended. Between the breaks people often asked him to finalize the project alone. And Thomas's viewpoint was that he was only collaborating to make it as perfect as possible. So that was never an option for him. Fortunately, Thomas's optimism was right! In mid

2011, the project was resurrected for that final push to completion.

The Physics

Thomas and I have worked well together and despite both being stubborn perfectionists I think we both have a sense of elegance. That's what I strive for, anyway -- and many times during our programming both of us have abandoned our own code for the more elegant code.

One of the last things to be added was "perfect" physics. There's a particular way that the boulders fall in the original game -- due to the scanning nature of the processing of the playfield. The playfield is scanned top to bottom, left to right, to find objects to move. To scan a playfield which is 40 x 20 characters in size -- that's 800 characters – the system has to load the character, decide what it is, do some processing based on that (i.e., boulders have to check if they need to fall, so they have to look at characters below and to the sides of them). That's thousands of loads from the playfield, just to get a single frame of game logic. Given each one of these (due to the difficult bank-switching involved) would take at the very least a few scan-lines of time (including the logic), then (very roughly) 1/3rd of a second is required per game loop. Too slow! So, the conclusion; it's not possible to scan the board. But every single other implementation of Boulder Dash scans the board - that's just how it's done!

But the Atari 2600 version is different! This is the most top-secret part of the whole project. This version of Boulder Dash does not scan the board and yet manages to have exactly the same physics as the original. But it was very difficult to achieve – and it was probably the last real bit of programming required for the game. There was a working physics engine early on, of course. But it was not correct physics. So when that last bit of code went in, and the physics started working 100% properly, that's the point at which Thomas and Andrew felt that the game would be finished.

Andrew's significant programming contribution to the game ended in early 2008. Thomas has been a major factor in driving the game to completion. In particular, "polish". One of those final bits of polish was the music on the title screen.

The Music

Luckily the weather during my 3 weeks summer vacation was absolutely terrible.

Digitized title music on an asymmetrical playfield with Cosmic Ark star-field and sprites - Thomas

With the awful weather over the summer vacation 2011, Thomas had a good excuse to work on the game almost all day long. Until then the game had no sound at all, so he took this task. When he was done, the game-play felt very different. With more senses involved, it had more depth and seemed more responsive. Both Thomas and Andrew were surprised what a big difference some little sound effects made. Music was also added to the title screen. This was a really big challenge and it took Thomas several weeks to complete. What a relief!

I doubt many people will ever understand what a truly amazing bit of tight coding this is. Firstly, we need to thank Fred Quimby for his initial music demo. This showed a technique which could produce the 'digitized' music that you hear in the final product. Fred's system was the first step in getting the music system interleaved with the title screen. It took Thomas weeks of work to rework the music system and mesh it with the title display. Given that the title is a complex asymmetrical playfield with embedded sprites and Cosmic Ark star-field, putting digitized music in with all that. It's amazing, and probably won't ever be matched.

The Agreement

In early 2004, Andrew contacted First Star Software to discuss the possibility of producing Boulder Dash on the Atari 2600. He negotiated a gentleman's agreement where demonstration videos of the progress of the Atari version could be shown on AtariAge, but no binaries or source code could be released.

First Star Software was very cautious – and understandably so - but with my promises to not release any binaries or source code they gave their blessing for me to show what I could do.

By honoring this agreement over nearly a decade, some degree of trust and mutual respect had been built between First Star Software, Thomas and Andrew. We had all shared the excitement of seeing Boulder Dash come alive on the Atari 2600, and all hoped for an eventual release. In 2011 all parties finally agreed on the terms of licensing and distribution for this 250-unit limited-edition.

The artwork was chosen as the winning entry of a competition held on AtariAge. Andrew then worked with André and Nathan to finalize the label and the box. Meanwhile Thomas and Andrew continued debugging and polishing the game, so that there are hopefully no major bugs left!

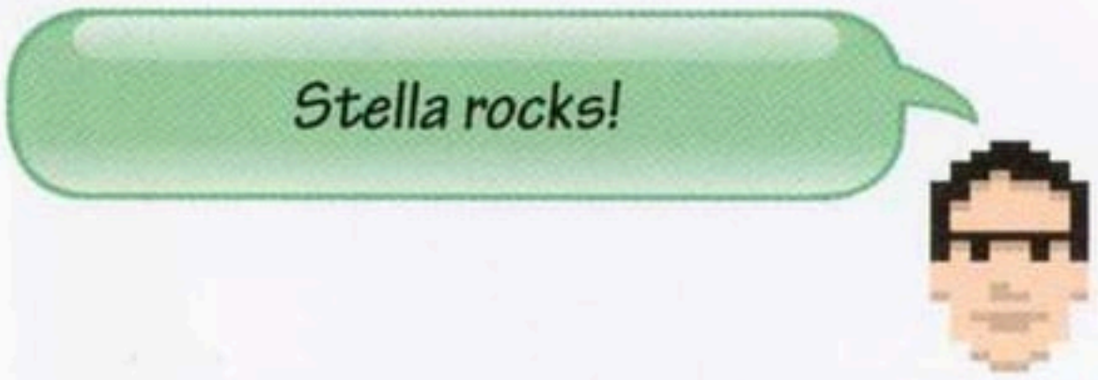
The End?

So, here we are in 2012 with the game finally on cartridge. FSS is a co-publisher with AtariAge – and we have one of the most complex Atari 2600 programs ever written... complete!

Acknowledgements

We have a number of people to thank for their important contributions to this game...

Stephen Anthony has for many years been the sole maintainer of Stella. He made it the #1 Atari 2600 emulator. In particular, the debugger became very helpful during development and saved us a lot of time. Thank you so much, Stephen!



Also our thanks go to Armin Vogl for the excellent Krokodile Cart which made it possible to test the game on real hardware, and who designed the 3E bank-switch scheme. To Fred Quimby, who provided the title game music demo and also designed and built the very special hardware required for Boulder Dash, our warmest thanks. Armin and Fred kindly supplied us with hardware development boards for testing on the real thing.

Richard M. Spitalny from First Star Software was a blast to work with. He has always been very professional, patient, and also very encouraging! Thank you Richard for hanging in there so long with us, and being so willing to negotiate.

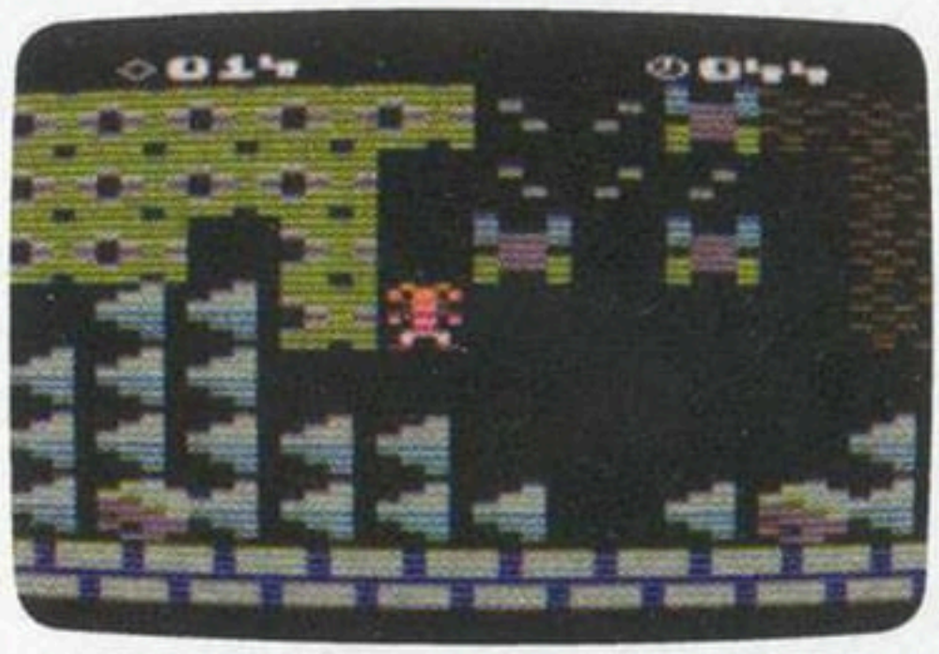
We're all thrilled with André Bolfing's unique competition-winning artwork adorning the box. It's something very special. Nathan Strum did his (usual) magic on the design of the manual, box and label. He has shown superhuman patience dealing with our endless "improvements" to the text and artwork. Our thanks, too, to Marc Oberhäuser, who turned design into reality by organizing the printing of the great box we hope you will always enjoy looking at on your shelf.

We must not forget to thank Albert Yarusso, who, even though always very busy, found the time to get this game into production and delivered to you.

Last, but not least, a big "THANK YOU!" to all those unnamed people who, for nearly 10 years, have continued to support this game. Each of you has helped make Boulder Dash for the Atari 2600 something we are very proud of!

FIRST STAR SOFTWARE INC®

BOULDER DASH®



The ultimate underground journey, but be forewarned - nothing less than perfection is required! Boulder Dash® will challenge your mind and reflexes like no other game!

Rockford™ digs feverishly, as boulders crash down all around him, through 16 mystical caves and 5 levels of difficulties. In his restless quest for gleaming jewels, Rockford works around walls of rock and avoids swirling fireflies. To win, he must turn his enemies into opportunities! He drops boulders through an enchanted wall, blocks the growing amoeba, transforms butterflies - magically turning them into precious stones! If he collects his required number of diamonds, the mysterious escape tunnel is revealed. After every 4 caves there is a playable intermission.

It will take all the strategy and thought you can muster to master the "physics" of Boulder Dash. Join Rockford and experience the excitement and beauty that awaits you.

- **For the Atari 2600 and 7800 consoles.**
- **Switchable TV formats (PAL and NTSC).**
- **For 1 or 2 players with 1 or 2 joysticks.**
- **16 caves, each with 5 challenging levels.**
- **4 playable intermission screens.**
- **Perfect physics and gameplay engine.**
- **Flicker-free, full-screen graphics.**
- **Amazing digital theme music.**
- **SaveKey support for multiple high scores.**

AtariAge