Subject: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 04 Aug 2019 22:15:50 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen (https://6502bench.com/) v1.2.0 has been released. This is just like v1.1, but implemented on a different UI library.

SourceGen v1.3.0-dev2 is now available. Key features:

- Partial support for ACME cross-assembler. I say "partial" because ACME's internal program counter is apparently only 16 bits, so 65816 code targeted outside bank zero is problematic.

- Apple /// SOS MLI calls, which are like ProDOS MLI calls but with BRK instead of JSR, are now handled. This change also breaks all existing SourceGen plugins, but I'm guessing there aren't a lot of those. (See https://github.com/fadden/6502bench/issues/44 for notes on updating existing projects..)

- Changed handling of BRK. I was treating it 65816-style, as a two-byte instruction, but hardly anything else does that, and things were getting awkward.

- Updated cc65 support for cc65 v2.18, although nothing much is different from v2.17 as far as SourceGen is concerned.

- Minor fixes to the new UI.

Source code and pre-built Windows binaries are available from
https://github.com/fadden/6502bench/releases

Subject: 6502bench SourceGen disassembler updated
Posted by Antoine Vignau on Mon, 05 Aug 2019 06:46:11 GMT
View Forum Message <> Reply to Message

Congrats!
av

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 05 Aug 2019 14:40:53 GMT
View Forum Message <> Reply to Message

Originally posted by: Frank M.

nice. thanks for adding acme. look forward to checking it out.

f

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 05 Aug 2019 20:31:46 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Monday, August 5, 2019 at 7:40:54 AM UTC-7, Frank M. wrote:
> nice. thanks for adding acme. look forward to checking it out.

ACME works great unless you're disassembling 65816 code. I've pinged the author to see if there's a way around the problem. (As noted in the docs, the internal PC is only 16 bits, so it has trouble with code outside bank zero.)

There's some potential room for improvement, e.g. using !xor to generate high-ASCII text strings.

Speaking of 65816 code, nobody can agree on what MVN/MVP args should look like: https://github.com/cc65/cc65/issues/925 . Merlin 32 is winning that one. :-)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 05 Aug 2019 21:45:22 GMT
View Forum Message <> Reply to Message

Originally posted by: Frank M.

On Monday, August 5, 2019 at 1:31:47 PM UTC-7, fadden wrote:
> There's some potential room for improvement, e.g. using !xor to generate high-ASCII text strings.

I was wondering how to do that. My first big project with acme targeted the][+ where I didn't need to worry about figuring it out.

f

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 07 Aug 2019 09:38:00 GMT
View Forum Message <> Reply to Message

Originally posted by: Brian Patrie

On 04/08/2019 17.15, fadden wrote:
> - Changed handling of BRK. I was treating it 65816-style, as a two-byte instruction, but hardly

anything else does that, and things were getting awkward.

AFAIK, The 6502 also treats BRK as a two byte instruction; but the weird precedent of treating it as a single was set (perhaps by the Apple II disassembler), and it stuck.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 07 Aug 2019 14:50:04 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Wednesday, August 7, 2019 at 2:38:03 AM UTC-7, Brian Patrie wrote:
> On 04/08/2019 17.15, fadden wrote:
>> - Changed handling of BRK. I was treating it 65816-style, as a two-byte instruction, but hardly anything else does that, and things were getting awkward.
>
> AFAIK, The 6502 also treats BRK as a two byte instruction; but the weird
> precedent of treating it as a single was set (perhaps by the Apple II
> disassembler), and it stuck.

It *behaves* like a two-byte instruction, but the data sheet for the 6502 says it's a 1-byte instruction. (See e.g. the table on page 8 of http://archive.6502.org/datasheets/synertek_sy6500_microproc essors_apr_1979.pdf)

The 65CE02 sheet correctly lists it as two bytes.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Thu, 08 Aug 2019 00:55:39 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

According to Rodnay Zaks, in "Programming the 6502" (page 126), BRK is a single byte instruction, with a value of zero ($00).

ftp://public.asimov.net/pub/apple_II/documentation/programmi ng/6502assembly/Programming the 6502_OCR.pdf

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Thu, 08 Aug 2019 00:57:51 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

According to Rodnay Zaks, in "Programming the 6502" (page 126), BRK is a single byte instruction, with a value of zero ($00).

ftp://public.asimov.net/pub/apple_II/documentation/programming/6502assembly/Programming%20the%206502_OCR.pdf

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 09 Sep 2019 18:25:16 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.3.0-dev5 is now available. There are three new features.

The first is local variable tables. When disassembling some code I ran into a part that did something like:

```
LDA ($00),Y
INC $02
```

And then later on used the same zero-page locations in a different way:

```
INC $00
LDA ($01),Y
```

This made assigning a global name useless. To make this better, you can now define tables at different points in the file, and create variable definitions. The same labels can be re-used, so one bit of code can refer to $00 as PTR and $02 as COUNT, while another bit can call $00 COUNT and $01 PTR.

This also works for the 8-bit operand to 65816 stack-relative instructions (LDA $00,S or LDA $(00,S),Y). So if you have compiled code that uses stack-relative operands to access arguments, you can put a table at the start of each function and give the arguments names. Local variables have an explicit width, so you can identify 2-byte and 3-byte pointers.

Appropriate code is generated for all four cross-assemblers. 64tass, cc65, and Merlin 32 use .var, .set, and ']', respectively. ACME uses !zone to constrain local variable scope.

The second feature is a rewrite of the way SourceGen handles strings. This allows the disassembler to tag strings as ASCII, C64 PETSCII, or C64 screen codes. Apple II code won't benefit from that directly, but because PETSCII strings sometimes have embedded control characters to do things like change the text color, I updated the string recognizers to accept some non-printable characters.

In practical terms, it means that what used to be shown like this:

```
  .STR 'HELLO, WORLD!'
  .DD1 $0D
```

is now the slightly nicer:

```
  .STR 'HELLO, WORLD!',$0D
```

I included $07, $0A, and $0D in the set for ASCII.

Having strings in various encodings is only helpful if you can tell them apart when reading the code, so string delimiters are now fully configurable. You can have Merlin-style single-vs-double quote for low vs. high ASCII, or define various prefix/suffix combinations. (I'm actually implementing support for high-ASCII immediate operands in non-Merlin assemblers by setting the suffix to "| $80".)

The third feature is a rewrite of the Edit Instruction Operand dialog, which had some mildly baffling "shortcuts" at the bottom. These have been replaced with buttons that let you edit labels, project symbols, and local variables associated with the operand.

Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 21 Sep 2019 23:30:56 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.3.0-alpha1 is now available. There is one new feature, and lots of little improvements.

SourceGen can now "export" the project listing to text, CSV, or HTML. This behaves similarly to the text-copy option that has been around for a while, in that it exports the text as it appears on the screen, rather than formatted for some specific assembler. This gives you full control over text delimiters, pseudo-opcodes, expression syntax, cycle counts, and so on.

As an exercise, I converted Bob Sander-Cederlof's disassembly of Applesoft to SourceGen format, and exported it to HTML. The project file and HTML output can be found here: https://bigflake.com/disasm/applesoft/

I'm not quite sold on the HTML style. All of the symbols are output as links, and the blue underlined text feels kind of distracting. I may tweak the formatting in the CSS file. Maybe even just make it look like plain text that happens to be clickable?

Source code and pre-built Windows executables for SourceGen are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Antoine Vignau on Sun, 22 Sep 2019 09:34:45 GMT
View Forum Message <> Reply to Message

Nice work, Andy!

av

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 23 Sep 2019 02:02:21 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Saturday, September 21, 2019 at 4:30:58 PM UTC-7, fadden wrote:
> As an exercise, I converted Bob Sander-Cederlof's disassembly of Applesoft to SourceGen format, and exported it to HTML. The project file and HTML output can be found here:
https://bigflake.com/disasm/applesoft/

I felt like getting more exercise.

Autostart ROM, direct from the Apple II Reference Manual:
https://bigflake.com/disasm/apple2-rom/F8ROM.html

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 25 Sep 2019 20:51:32 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Sunday, September 22, 2019 at 7:02:22 PM UTC-7, fadden wrote:
> On Saturday, September 21, 2019 at 4:30:58 PM UTC-7, fadden wrote:
>> As an exercise, I converted Bob Sander-Cederlof's disassembly of Applesoft to SourceGen format, and exported it to HTML. The project file and HTML output can be found here:
https://bigflake.com/disasm/applesoft/
>
> I felt like getting more exercise.
>
> Autostart ROM, direct from the Apple II Reference Manual:
> https://bigflake.com/disasm/apple2-rom/F8ROM.html

Strange, this last link is not working directly from Google Groups with MSIE-11 (error message: this page cannot be displayed), but the address works if I copy it to the IE address bar and go to it from there.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Fri, 27 Sep 2019 21:54:48 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen (https://6502bench.com/) v1.3.0 has been released. Key changes since v1.2:

- Added local variable tables. These are redefinable symbols that can be used for zero-page and stack-relative operands.
- Added support for the ACME cross-assembler (65816 support is limited).
- Added support for multiple character encodings. C64 PETSCII and screen code strings can be detected automatically, and the encodings may now be manually specified for characters and strings. Some non-printable characters, such as CR/LF, are now allowed in strings.
- Added ability to "export" the code listing to text, CSV, or HTML.
- Character and string delimiters are configurable.
- Rewrote the Edit Instruction Operand dialog to make the "shortcuts" less obscure.
- Made various improvements to source code generation for 64tass, cc65 (now v2.18), and Merlin 32.
- Changed handling of BRK to treat it as a single-byte instruction with no operand.
- Added support for Apple /// SOS MLI calls, which use BRK instructions with inline data.
- Various bug fixes and miscellaneous improvements.

Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Fri, 27 Sep 2019 22:04:18 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Sunday, September 22, 2019 at 7:02:22 PM UTC-7, fadden wrote:
> I felt like getting more exercise.

Added another one. This is the one that inspired me to add local variable tables: Bill Budge's 3-D Graphics System and Game Tool. I disassembled a "module" generated by the tool to see how he handled math and line drawing.

https://bigflake.com/disasm/apple2-budge3d/

(if you're not familiar with this program, I've got a bunch of links on the page, including a video of the demo that comes with it)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 02 Oct 2019 07:41:43 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Friday, September 27, 2019 at 2:54:50 PM UTC-7, fadden wrote:
> 6502bench SourceGen (https://6502bench.com/) v1.3.0 has been released. Key changes since v1.2:
>
> - Added local variable tables. These are redefinable symbols that can be used for zero-page and stack-relative operands.
> - Added support for the ACME cross-assembler (65816 support is limited).
> - Added support for multiple character encodings. C64 PETSCII and screen code strings can be detected automatically, and the encodings may now be manually specified for characters and strings. Some non-printable characters, such as CR/LF, are now allowed in strings.
> - Added ability to "export" the code listing to text, CSV, or HTML.
> - Character and string delimiters are configurable.
> - Rewrote the Edit Instruction Operand dialog to make the "shortcuts" less obscure.
> - Made various improvements to source code generation for 64tass, cc65 (now v2.18), and Merlin 32.
> - Changed handling of BRK to treat it as a single-byte instruction with no operand.
> - Added support for Apple /// SOS MLI calls, which use BRK instructions with inline data.
> - Various bug fixes and miscellaneous improvements.
>
> Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

For those of us who like to double-click a project file and have it run the SourceGen program with the file loaded, here is a sameple RegEdit file to adapt and merge [for *.dis65 & *.sym65 files (on W7U64DTC)]:

Windows Registry Editor Version 5.00

```
 ;============================================================ ===
; SourceGen Files                          (SourceGen.reg)
 ;============================================================ ===


 ;============================================================ ===
; Notes:
 ;============================================================ ===
;
; Unzipped "6502bench130.zip" to "C:\Program Files\6502bench"
; (on a Windows 7 Ultimate (64-bit) Lenovo desktop computer).
;
 ;------------------------------------------------------------ ---
;
; On first OpenWith, named:
```

```
;
; "*.dis65" files, "SourceGen Project File"
;                (Open with: "SourceGen.exe"); and,
;
; "*.sym65" files, "SourceGen Platform Symbol File"
;                (Open with: "NotePad.exe").
;
 ;------------------------------------------------------- ---
;
; Renamed *_auto_files from "*65_auto_file" to "*65file".
;
 ;======================================================== ===


 ;======================================================== ===

[HKEY_CURRENT_USER\Software\Classes\.dis65]
@="dis65file"

 ;------------------------------------------------------- ---

 [HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion \Explorer\FileExts\.dis65]
 [HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion
\Explorer\FileExts\.dis65\OpenWithList]
"a"="SourceGen.exe"
"MRUList"="a"

 [HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion
\Explorer\FileExts\.dis65\OpenWithProgids]
"dis65file"=hex(0):

 ;------------------------------------------------------- ---

[-HKEY_CURRENT_USER\Software\Classes\dis65_auto_file]

[HKEY_CURRENT_USER\Software\Classes\dis65file]
@="SourceGen Project File"

[HKEY_CURRENT_USER\Software\Classes\dis65file\shell]
[HKEY_CURRENT_USER\Software\Classes\dis65file\shell\open]
 [HKEY_CURRENT_USER\Software\Classes\dis65file\shell\open\com mand]
@="\"C:\\Program Files\\6502bench\\SourceGen.exe\" \"%1\""

 ;======================================================== ===

[HKEY_CURRENT_USER\Software\Classes\.sym65]
@="sym65file"
```

```
 ;------------------------------------------------------- ---

 [HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion
\Explorer\FileExts\.sym65]
 [HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion
\Explorer\FileExts\.sym65\OpenWithList]
"a"="NOTEPAD.EXE"
"MRUList"="a"

 [HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion
\Explorer\FileExts\.sym65\OpenWithProgids]
"sym65file"=hex(0):

 ;------------------------------------------------------- ---

[-HKEY_CURRENT_USER\Software\Classes\sym65_auto_file]

[HKEY_CURRENT_USER\Software\Classes\sym65file]
;@="SourceGen Platform Symbol File"
@="SourceGen Symbol File"; (short name)

[HKEY_CURRENT_USER\Software\Classes\sym65file\shell]
[HKEY_CURRENT_USER\Software\Classes\sym65file\shell\edit]
 [HKEY_CURRENT_USER\Software\Classes\sym65file\shell\edit\com mand]
;@="%SystemRoot%\\system32\\NOTEPAD.EXE %1"
 @=hex(2):25,00,53,00,79,00,73,00,74,00,65,00,6d,00,52,00,6f, 00,6f,00,74,00,25,\
   00,5c,00,73,00,79,00,73,00,74,00,65,00,6d,00,33,00,32,00,5c, 00,4e,00,4f,00,\
   54,00,45,00,50,00,41,00,44,00,2e,00,45,00,58,00,45,00,20,00, 25,00,31,00,00,\
  00

[HKEY_CURRENT_USER\Software\Classes\sym65file\shell\open]
 [HKEY_CURRENT_USER\Software\Classes\sym65file\shell\open\com mand]
;@="%SystemRoot%\\system32\\NOTEPAD.EXE %1"
 @=hex(2):25,00,53,00,79,00,73,00,74,00,65,00,6d,00,52,00,6f, 00,6f,00,74,00,25,\
   00,5c,00,73,00,79,00,73,00,74,00,65,00,6d,00,33,00,32,00,5c, 00,4e,00,4f,00,\
   54,00,45,00,50,00,41,00,44,00,2e,00,45,00,58,00,45,00,20,00, 25,00,31,00,00,\
  00

 ;======================================================== ===

 [HKEY_CURRENT_USER\Software\Classes\Applications\SourceGen.e xe]
 [HKEY_CURRENT_USER\Software\Classes\Applications\SourceGen.e xe\shell]
 [HKEY_CURRENT_USER\Software\Classes\Applications\SourceGen.e xe\shell\open]
 [HKEY_CURRENT_USER\Software\Classes\Applications\SourceGen.e xe\shell\open\command]
@="\"C:\\Program Files\\6502bench\\SourceGen.exe\" \"%1\""

 ;======================================================== ===
```

[HKEY_CURRENT_USER\Software\Classes\Local Settings\Software\Microsoft\Windows\Shell\MuiCache]
"C:\\Program Files\\6502bench\\SourceGen.exe"="SourceGen"

;=== ===

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 02 Oct 2019 14:58:06 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Wednesday, October 2, 2019 at 12:41:44 AM UTC-7, James Davis wrote:
> For those of us who like to double-click a project file and have it run the SourceGen program with the file loaded, here is a sameple RegEdit file to adapt and merge [for *.dis65 & *.sym65 files (on W7U64DTC)]:

Nice!

In related news, I found a crashing bug, so v1.3.1 has been released. (Open a project, and immediately click on the "Actions" menu... boom. Whoops. Edge case where no lines are selected.)

In related news, I can recommend the "OneFlow" methodology for managing hotfixes in a simple git repository.

FWIW, v1.3 added a crash trap that writes a "CrashLog.txt" into the installation directory when a fatal exception is encountered. So if the program crashes, look there for details.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Thu, 03 Oct 2019 06:13:50 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Wednesday, October 2, 2019 at 7:58:08 AM UTC-7, fadden wrote:
> On Wednesday, October 2, 2019 at 12:41:44 AM UTC-7, James Davis wrote:
>> For those of us who like to double-click a project file and have it run the SourceGen program with the file loaded, here is a sameple RegEdit file to adapt and merge [for *.dis65 & *.sym65 files (on W7U64DTC)]:
>
> Nice!
>
>

> In related news, I found a crashing bug, so v1.3.1 has been released. (Open a project, and immediately click on the "Actions" menu... boom. Whoops. Edge case where no lines are selected.)
>
> In related news, I can recommend the "OneFlow" methodology for managing hotfixes in a simple git repository.
>
> FWIW, v1.3 added a crash trap that writes a "CrashLog.txt" into the installation directory when a fatal exception is encountered. So if the program crashes, look there for details.

Andy:

That happened to me, too. Do you want me to email the crash report to you?

James Davis

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Thu, 03 Oct 2019 14:37:25 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Wednesday, October 2, 2019 at 11:13:52 PM UTC-7, James Davis wrote:
> That happened to me, too. Do you want me to email the crash report to you?

If the crash looks like the one in https://github.com/fadden/6502bench/issues/48 (look for "CanCreateLocalVariableTable()" a few lines from the top), then it's already fixed in v1.3.1.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 05 Oct 2019 08:20:22 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Thursday, October 3, 2019 at 7:37:27 AM UTC-7, fadden wrote:
> On Wednesday, October 2, 2019 at 11:13:52 PM UTC-7, James Davis wrote:
>> That happened to me, too. Do you want me to email the crash report to you?
>
> If the crash looks like the one in https://github.com/fadden/6502bench/issues/48 (look for "CanCreateLocalVariableTable()" a few lines from the top), then it's already fixed in v1.3.1.

I have found two crashlog.txt files (that I renamed):

6502bench SourceGen Disassembler CrashLog [2019-10-01].txt (13KB) [Contains "CanCreateLocalVariableTable()"]

6502bench SourceGen Disassembler CrashLog [2019-10-03].txt (5KB) [Does not contain "CanCreateLocalVariableTable()"]

They do not say internally which version of 6502bench SourceGen crashed. They are Greek to me. If you want to look at them, I will send them to you. If you do not, I will delete them.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 05 Oct 2019 08:40:15 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

Andy,

I have gone through the Tutorials and have found a few quirks, but it is getting too late (1:35 AM), so I will write about it the next time I am online here.

Good night,

James Davis

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 05 Oct 2019 18:23:43 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Saturday, October 5, 2019 at 1:20:24 AM UTC-7, James Davis wrote:
> I have found two crashlog.txt files (that I renamed):
>
> 6502bench SourceGen Disassembler CrashLog [2019-10-01].txt (13KB) [Contains "CanCreateLocalVariableTable()"]
>
> 6502bench SourceGen Disassembler CrashLog [2019-10-03].txt (5KB) [Does not contain "CanCreateLocalVariableTable()"]

The second one sounds interesting. Please do forward that to my e-mail if you get a chance. (Alternatively, create a new issue at https://github.com/fadden/6502bench/issues and attach the file.)

> They do not say internally which version of 6502bench SourceGen crashed.

They do now. :-) When you asked about the logs a few days back, it occurred to me that the crash logs should do a better job of identifying themselves, so now the app version and OS

version are printed before each crash. (This'll start showing up in the first 1.4 dev release.)

Note that crashes are appended to CrashLog.txt if it already exists. Search for "***" to find the start of each. Your files are small enough that they probably only have one crash in them (WPF generates some deep stack traces).

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 06 Oct 2019 08:01:36 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Saturday, October 5, 2019 at 1:40:17 AM UTC-7, James Davis wrote:
> Andy,
>
> I have gone through the Tutorials and have found a few quirks, but it is getting too late (1:35 AM), so I will write about it the next time I am online here.
>
> Good night,
>
> James Davis

Here are some of the quirks I have noticed ...

 ========== ====
6502bench SourceGen Quirks:
 ========== ====

In the disassembly window, at the bottom, when the slider bar is not all the way to the left, and/or any amount or all the way to the right, double-clicking on a column (e.g., the label-column*) brings up the action-window for the column just to the right of the column double-clicked (e.g., the attribute-column*, respectively); unless the cursor is an equal offset to the right (within the column-field selected) as the distance (lighter-area) to the right of the slider-bar. [In other words, (it's "Weird!" that) you have to offset your mouse-click, too!]

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

In the disassembly window, I keep expecting the (long or short) comments and notes to work like in a word-processor (e.g., NotePad, MetaPad or {my preferred} WordPad), by clicking in the column or field once, then trying to place the cursor within the field by clicking it again somewhere else within the field; instead of quickly double-clicking it in the first place.

It would be nice if comments and notes also worked like in a word-processor or worked to bring up the edit-window with a second single-click (on new short-comments, if possible, and existing comments {both, within the comment-column}, and on long-comments and notes).

A quick double-click should not be necessary for editing existing (long or short) comments and notes; two single-clicks should also work within the same field {of lines--for "multi-line" comments and notes} (no matter where the mouse/cursor has been between single-clicks). [Some kind of "field-single-clicked-twice (with no-clicks-outside-the-field-between-clicks)" flag might do the trick.]

There needs to be a way [using Windows' standard (large) Color selector dialog] to set some or all the colors used in the application to user preferred colors (e.g., all the different highlighting colors, window background color, etc.) vs. the Users' windows-theme or Windows' standard colors; one or the other of which should be selectable as a base to start with (or to build upon).

[See attached picture.]

[For example, I would prefer all the 'SELECTED' field-highlighting to be the same (e.g., dark background with light lettering {e.g., blue/white}), not the (light background and dark lettering) colors you have enabled for long-comments and notes.--{No offense meant!}]

Allow fonts (for individual words and characters) to be normal, bold, italic, underlined, struck-through, and subscripted or superscripted; like in a word-processor (e.g., WordPad {RTF}); especially in comments and notes. [Maybe there is an RTF.DLL for textual fields you could incorporate.]

In the disassembly window, at the top, the column headers should also be re-adjustable using the standard Windows ctrl-+, shift-+, and non-standard ctrl-shift-+ key-combinations; to set them to (ctrl-+) the maximum column widths used (below them), and (shift-+) to the same but with the column header-name widths as a minimum (so the names are not truncated), and (ctrl-shift-+)

to reset them to their application defaults; respectively.

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

It would be nice to be able to insert extra blank-lines, long-comments and notes, above and/or below other lines and at the very end, in the disassemblies.

And/or, it would be nice to be able to drag-&-drop blank-lines, long-comments and notes, up or down in the disassemblies.

A line space is achievable as a single space in a long-comment or note, but they have inline artifacts!

To put anything below the last line, one would have to pad the original binary/hexadecimal machine code with an extra zero byte, and remove it after editing the disassembly. Would the edit remain or disappear? I will have to test this.

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

It would be nice to be able to load an ASCII text of another disassembly created by other means and have SourceGen be able to use it (detecting which assembler format to use, too), with or without a binary/hexadecimal machine code file to go with it.

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

The settings keep resetting/reverting to 64tass on the last three tabs. They do not stay set to Merlin from session to session. Do you think this is because of the windows permissions for the path I am using, "C:\Program Files\6502bench \"? Or, could it be a bug in SourceGen?

========== ====

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 06 Oct 2019 21:12:11 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Sunday, October 6, 2019 at 1:01:38 AM UTC-7, James Davis wrote:
> In the disassembly window, at the bottom, when the slider bar is
> not all the way to the left, and/or any amount or all the way to
> the right, double-clicking on a column (e.g., the label-column*)
> brings up the action-window for the column just to the right of

> the column double-clicked (e.g., the attribute-column*,
> respectively);

That's a bug. WPF (Windows Presentation Foundation) is based on the philosophy that hard things should be possible, and easy things should be really hard. One manifestation of this is the absence of "what row and column was clicked on" methods in ListView and DataGrid. My implementation failed to take the horizontal scroll position into account.

Fixed.

> In the disassembly window, I keep expecting the (long or short)
> comments and notes to work like in a word-processor (e.g.,
> NotePad, MetaPad or {my preferred} WordPad)

I have a "to do" list item for in-place editing of labels and comments. There might be a way to convince WPF to do it for me, but it'll likely require using a fully custom control.

The main disassembly window is a WPF ListView control with some style changes, mostly to have text that extends across multiple columns. This was sufficiently difficult that I created a separate github project just to play with it (https://github.com/fadden/DisasmUiTest). I'm not yet experienced with WPF to know how hard in-place editing would be.

> A quick double-click should not be necessary for editing
> existing (long or short) comments and notes; two single-clicks
> should also work within the same field {of lines--for "multi-
> line" comments and notes} (no matter where the mouse/cursor has
> been between single-clicks).

I think that would make sense for in-place editing. I'm picturing the way Windows lets you rename a file by selecting it, and then doing a second single click. I don't think it feels right when the action is to open an editor window though.

> There needs to be a way [using Windows' standard (large) Color
> selector dialog] to set some or all the colors used in the
> application to user preferred colors (e.g., all the different
> highlighting colors, window background color, etc.) vs. the
> Users' windows-theme or Windows' standard colors; one or the
> other of which should be selectable as a base to start with (or
> to build upon).

I have a long-standing request for a "dark" theme. Seems like it should be straightforward in WPF, but I haven't figured it out yet.

> [For example, I would prefer all the 'SELECTED' field-
> highlighting to be the same (e.g., dark background with light
> lettering {e.g., blue/white}), not the (light background and
> dark lettering) colors you have enabled for long-comments and
> notes.--{No offense meant!}]

That's another bug. The custom style that allows me to have long comments extend across multiple columns also requires me to define how highlighting works. In Win10, the system-wide theme has essentially no effect on the app, so my selected colors look the same as everything else in the list. I tried it just now on Win7, and the "high contrast" system theme significantly altered all of the colors in SourceGen, except for the ones in the replacement style. Ugh.

So I need to figure out how to get the appropriate color out of the system style. There's some chance that the Win7 Aero stuff behaves in a totally different way from what Win10 uses, which would make this even more fun.

> Allow fonts (for individual words and characters) to be normal,
> bold, italic, underlined, struck-through, and subscripted or
> superscripted; like in a word-processor (e.g., WordPad {RTF});
> especially in comments and notes.

This is doable, but I'm a little hesitant because one of the primary goals of SourceGen is to generate plain text source code, which means either stripping the formatting (which could lose information, e.g. the fact that text was struck out) or generating comments with RTF format strings (which is not especially easy for humans to read). It's useful for HTML output, but I'd rather have people express themselves within the confines of a fixed-width Unicode font so that things look the same everywhere.

> In the disassembly window, at the top, the column headers should
> also be re-adjustable using the standard Windows ctrl-+, shift-
> +, and non-standard ctrl-shift-+ key-combinations

I'm unclear as to how standard those are in terms of column widths. For example, in Chrome, those change the page "zoom" level.

> It would be nice to be able to insert extra blank-lines, long-
> comments and notes, above and/or below other lines and at the
> very end, in the disassemblies.
>
> And/or, it would be nice to be able to drag-&-drop blank-lines,
> long-comments and notes, up or down in the disassemblies.

This is doable. The reason it's not done now is that everything is associated with a file offset, and you only get one of each thing. So if you add a long comment, that comment is associated with a specific byte in the file, and is always displayed above that byte.

The mechanism could be made more general, e.g. multiple comments per offset, which can be placed above or below. Blank lines can be fiddled with. I think drag & drop or some other affordance would be necessary, since tweaking with the cosmetic aspects is awkward if you have to do it through a dialog window.

> A line space is achievable as a single space in a long-comment
> or note, but they have inline artifacts!

It would be easy to treat an empty single-line long comment as an explicit blank line. There's a potential issue in that it's hard to tell it apart from the automatically generated blank lines. (I could have a "don't put leading semicolons on blank lines" checkbox.)

> To put anything below the last line, one would have to pad the
> original binary/hexadecimal machine code with an extra zero
> byte, and remove it after editing the disassembly. Would the
> edit remain or disappear? I will have to test this.

The .dis65 file includes the file length and CRC, so you'd have to update those values before SourceGen would accept the altered file. It would then notice that you have something with an invalid offset, and ignore it. So.... don't do this.

> It would be nice to be able to load an ASCII text of another
> disassembly created by other means and have SourceGen be able to
> use it (detecting which assembler format to use, too), with or
> without a binary/hexadecimal machine code file to go with it.

This would be best done with a tool that generated a .dis65 file. The goal of such a thing would be to facilitate conversion of source code from one assembler to another by using SourceGen as a least-common-denominator format. The trouble is that SourceGen is far less expressive than a typical assembler, and you'd lose a lot. But if you can keep most of the labels and comments, that'd be an easier place to start from.

> The settings keep resetting/reverting to 64tass on the last
> three tabs. They do not stay set to Merlin from session to
> session.

This isn't a bug so much as it's just bad UI design. It annoys me too.

If you launch SourceGen v1.0, the "Quick Set" box has three buttons, "Default", "cc65", and "Merlin 32". You click the button, and the various fields get reconfigured. Simple.

In SourceGen v1.1, I added support for 64tass, and realized that my previous scheme didn't scale well, and made it difficult to add new assemblers. So instead of a pair of buttons, you have a pop-up menu, with a "Set" button to actually apply the values.

The problem, as you have noted, is that it looks like something is configured for 64tass, even though in practice nothing happens when you select a different assembler from the pop-up menu.

So the question is how to change the UI to make sense. One possibility is to get rid of "Set" and just make the combo box live: selecting a different assembler immediately updates all values. A new entry, "Custom", would hold the state of any recent edits. The pop-up would either show the name of the assembler (if all fields match) or "custom" (if any don't). So as soon as you make a change to any field, it switches to "custom" and obliterates any previously established custom setting... but I don't see a way around doing so that doesn't require wrapping a heavyweight customization UI around something that should be simple.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 07 Oct 2019 01:38:46 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

SourceGen v1.3.2 has been released, to fix an "export HTML" crash that is likely responsible for the [2019-10-03] CrashLog.txt. I also threw in the fix for selecting the wrong column when the disassembly window is scrolled horizontally.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 00:36:11 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

Two more quirks (but they are probably related to each other):

1. *.dis65 files do not work (reload into SourceGen) after being renamed (e.g., "Apple2.ROM.dis65" to "Apple2Plus.ROM.dis65" {SourceGen says it does not exist!}).

2. *.dis65 files do not work (reload into SourceGen) within a folder named "Apple II+"; but they will work after renaming the folder to "Apple II Plus" (e.g., "Apple2.ROM.dis65" does not work from within the "Apple II+" folder, but works in the "Apple II Plus" folder (renamed); but "Apple2Plus.ROM.dis65" still does not work {SourceGen says it does not exist!}).

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 00:53:45 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Monday, October 7, 2019 at 5:36:13 PM UTC-7, James Davis wrote:
> 1. *.dis65 files do not work (reload into SourceGen) after being renamed (e.g.,
> "Apple2.ROM.dis65" to "Apple2Plus.ROM.dis65" {SourceGen says it does not
> exist!}).
>
> 2. *.dis65 files do not work (reload into SourceGen) within a folder named
> "Apple II+"; but they will work after renaming the folder to "Apple II Plus"
[...]

I tried this a few different ways and didn't have any problems, so we're apparently doing different things. I will point out that the "Recent Projects" menu item, and the two recent project buttons on

the initial screen, do not update when files move around in the filesystem. So if you rename or relocate a project, you will need to File > Open it rather than using the quick-load button.

This can be a little confusing because the buttons on the main screen only show the filename, so if you've opened something in two different places it's not immediately obvious which one is which. The Recent Projects menu does show full paths, so check and see if that matches up.

(It now occurs to me that the existing UI is dumb, because it's showing you a quick-load button for a project that isn't there. I'll fix that.)

The only requirement on filenames should be that the .dis65 file and the data file live in the same directory, and have the same name (just without ".dis65" for the data file). If that's not the case you'll get the opportunity to locate the file manually. Otherwise you should be able to call it anything Windows will tolerate.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 00:58:07 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

I am not using Aero effects. I am using Windows 7 Ultimate 64-bit made to look like Classic Windows 95~98se with a Classic Start Menu. [I am a square!-box/ASCII-text kind of guy (;-D). But I do like WordPad and RTF, too.]

My preferred Apple II disassembler was the Mini-Assembler/Disassembler. And, my preferred Apple II assembler was EDASM. So, I suppose you should just stick to ASCII text for comments and notes. I prefer the Merlin format in SourceGen because it is closest to EDASM. Can you add an EDASM format? It is very similar to the original Apple II Mini-Assembler/Disassembler format.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 00:58:38 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Sunday, October 6, 2019 at 2:12:12 PM UTC-7, fadden wrote:
> On Sunday, October 6, 2019 at 1:01:38 AM UTC-7, James Davis wrote:
>> In the disassembly window, at the bottom, when the slider bar is
>> not all the way to the left, and/or any amount or all the way to
>> the right, double-clicking on a column (e.g., the label-column*)
>> brings up the action-window for the column just to the right of
>> the column double-clicked (e.g., the attribute-column*,
>> respectively);
>

> That's a bug. WPF (Windows Presentation Foundation) is based on the philosophy that hard things should be possible, and easy things should be really hard. One manifestation of this is the absence of "what row and column was clicked on" methods in ListView and DataGrid. My implementation failed to take the horizontal scroll position into account.
>
> Fixed.
>
>> In the disassembly window, I keep expecting the (long or short)
>> comments and notes to work like in a word-processor (e.g.,
>> NotePad, MetaPad or {my preferred} WordPad)
>
> I have a "to do" list item for in-place editing of labels and comments. There might be a way to convince WPF to do it for me, but it'll likely require using a fully custom control.
>
> The main disassembly window is a WPF ListView control with some style changes, mostly to have text that extends across multiple columns. This was sufficiently difficult that I created a separate github project just to play with it (https://github.com/fadden/DisasmUiTest). I'm not yet experienced with WPF to know how hard in-place editing would be.
>
>> A quick double-click should not be necessary for editing
>> existing (long or short) comments and notes; two single-clicks
>> should also work within the same field {of lines--for "multi-
>> line" comments and notes} (no matter where the mouse/cursor has
>> been between single-clicks).
>
> I think that would make sense for in-place editing. I'm picturing the way Windows lets you rename a file by selecting it, and then doing a second single click. I don't think it feels right when the action is to open an editor window though.
>
>> There needs to be a way [using Windows' standard (large) Color
>> selector dialog] to set some or all the colors used in the
>> application to user preferred colors (e.g., all the different
>> highlighting colors, window background color, etc.) vs. the
>> Users' windows-theme or Windows' standard colors; one or the
>> other of which should be selectable as a base to start with (or
>> to build upon).
>
> I have a long-standing request for a "dark" theme. Seems like it should be straightforward in WPF, but I haven't figured it out yet.
>
>> [For example, I would prefer all the 'SELECTED' field-
>> highlighting to be the same (e.g., dark background with light
>> lettering {e.g., blue/white}), not the (light background and
>> dark lettering) colors you have enabled for long-comments and
>> notes.--{No offense meant!}]
>
> That's another bug. The custom style that allows me to have long comments extend across multiple columns also requires me to define how highlighting works. In Win10, the system-wide

theme has essentially no effect on the app, so my selected colors look the same as everything else in the list. I tried it just now on Win7, and the "high contrast" system theme significantly altered all of the colors in SourceGen, except for the ones in the replacement style. Ugh.
>
> So I need to figure out how to get the appropriate color out of the system style. There's some chance that the Win7 Aero stuff behaves in a totally different way from what Win10 uses, which would make this even more fun.
>
>> Allow fonts (for individual words and characters) to be normal,
>> bold, italic, underlined, struck-through, and subscripted or
>> superscripted; like in a word-processor (e.g., WordPad {RTF});
>> especially in comments and notes.
>
> This is doable, but I'm a little hesitant because one of the primary goals of SourceGen is to generate plain text source code, which means either stripping the formatting (which could lose information, e.g. the fact that text was struck out) or generating comments with RTF format strings (which is not especially easy for humans to read). It's useful for HTML output, but I'd rather have people express themselves within the confines of a fixed-width Unicode font so that things look the same everywhere.
>
>> In the disassembly window, at the top, the column headers should
>> also be re-adjustable using the standard Windows ctrl-+, shift-
>> +, and non-standard ctrl-shift-+ key-combinations
>
> I'm unclear as to how standard those are in terms of column widths. For example, in Chrome, those change the page "zoom" level.
>
>> It would be nice to be able to insert extra blank-lines, long-
>> comments and notes, above and/or below other lines and at the
>> very end, in the disassemblies.
>>
>> And/or, it would be nice to be able to drag-&-drop blank-lines,
>> long-comments and notes, up or down in the disassemblies.
>
> This is doable. The reason it's not done now is that everything is associated with a file offset, and you only get one of each thing. So if you add a long comment, that comment is associated with a specific byte in the file, and is always displayed above that byte.
>
> The mechanism could be made more general, e.g. multiple comments per offset, which can be placed above or below. Blank lines can be fiddled with. I think drag & drop or some other affordance would be necessary, since tweaking with the cosmetic aspects is awkward if you have to do it through a dialog window.
>
>> A line space is achievable as a single space in a long-comment
>> or note, but they have inline artifacts!
>
> It would be easy to treat an empty single-line long comment as an explicit blank line. There's a potential issue in that it's hard to tell it apart from the automatically generated blank lines. (I could

have a "don't put leading semicolons on blank lines" checkbox.)
>
>> To put anything below the last line, one would have to pad the
>> original binary/hexadecimal machine code with an extra zero
>> byte, and remove it after editing the disassembly. Would the
>> edit remain or disappear? I will have to test this.
>
> The .dis65 file includes the file length and CRC, so you'd have to update those values before SourceGen would accept the altered file. It would then notice that you have something with an invalid offset, and ignore it. So.... don't do this.
>
>> It would be nice to be able to load an ASCII text of another
>> disassembly created by other means and have SourceGen be able to
>> use it (detecting which assembler format to use, too), with or
>> without a binary/hexadecimal machine code file to go with it.
>
> This would be best done with a tool that generated a .dis65 file. The goal of such a thing would be to facilitate conversion of source code from one assembler to another by using SourceGen as a least-common-denominator format. The trouble is that SourceGen is far less expressive than a typical assembler, and you'd lose a lot. But if you can keep most of the labels and comments, that'd be an easier place to start from.
>
>> The settings keep resetting/reverting to 64tass on the last
>> three tabs. They do not stay set to Merlin from session to
>> session.
>
> This isn't a bug so much as it's just bad UI design. It annoys me too.
>
> If you launch SourceGen v1.0, the "Quick Set" box has three buttons, "Default", "cc65", and "Merlin 32". You click the button, and the various fields get reconfigured. Simple.
>
> In SourceGen v1.1, I added support for 64tass, and realized that my previous scheme didn't scale well, and made it difficult to add new assemblers. So instead of a pair of buttons, you have a pop-up menu, with a "Set" button to actually apply the values.
>
> The problem, as you have noted, is that it looks like something is configured for 64tass, even though in practice nothing happens when you select a different assembler from the pop-up menu.
>
> So the question is how to change the UI to make sense. One possibility is to get rid of "Set" and just make the combo box live: selecting a different assembler immediately updates all values. A new entry, "Custom", would hold the state of any recent edits. The pop-up would either show the name of the assembler (if all fields match) or "custom" (if any don't). So as soon as you make a change to any field, it switches to "custom" and obliterates any previously established custom setting... but I don't see a way around doing so that doesn't require wrapping a heavyweight customization UI around something that should be simple.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 01:14:35 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Monday, October 7, 2019 at 5:53:46 PM UTC-7, fadden wrote:
> On Monday, October 7, 2019 at 5:36:13 PM UTC-7, James Davis wrote:
>> 1. *.dis65 files do not work (reload into SourceGen) after being renamed (e.g.,
>> "Apple2.ROM.dis65" to "Apple2Plus.ROM.dis65" {SourceGen says it does not
>> exist!}).
>>
>> 2. *.dis65 files do not work (reload into SourceGen) within a folder named
>> "Apple II+"; but they will work after renaming the folder to "Apple II Plus"
> [...]
>
>
> I tried this a few different ways and didn't have any problems, so we're apparently doing different things. I will point out that the "Recent Projects" menu item, and the two recent project buttons on the initial screen, do not update when files move around in the filesystem. So if you rename or relocate a project, you will need to File > Open it rather than using the quick-load button.
>
> This can be a little confusing because the buttons on the main screen only show the filename, so if you've opened something in two different places it's not immediately obvious which one is which. The Recent Projects menu does show full paths, so check and see if that matches up.
>
> (It now occurs to me that the existing UI is dumb, because it's showing you a quick-load button for a project that isn't there. I'll fix that.)
>
> The only requirement on filenames should be that the .dis65 file and the data file live in the same directory, and have the same name (just without ".dis65" for the data file). If that's not the case you'll get the opportunity to locate the file manually. Otherwise you should be able to call it anything Windows will tolerate.

No, that's not it. I am talking about double-clicking the *.dis65 files to start a fresh instance of SourceGen (e.g., with my reg-edits {submitted above}) [with no other instances of SourceGen running]. It must have something to do with the "+" in the folder name, the double-dots in the filenames, long filenames, or long pathnames (my path/filename is pretty long). It is probably MicroSoft-Windows' fault!

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 01:19:15 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Monday, October 7, 2019 at 6:14:37 PM UTC-7, James Davis wrote:
> On Monday, October 7, 2019 at 5:53:46 PM UTC-7, fadden wrote:
>> On Monday, October 7, 2019 at 5:36:13 PM UTC-7, James Davis wrote:
>>> 1. *.dis65 files do not work (reload into SourceGen) after being renamed (e.g.,
>>> "Apple2.ROM.dis65" to "Apple2Plus.ROM.dis65" {SourceGen says it does not
>>> exist!}).
>>>
>>> 2. *.dis65 files do not work (reload into SourceGen) within a folder named
>>> "Apple II+"; but they will work after renaming the folder to "Apple II Plus"
>> [...]
>>
>>
>> I tried this a few different ways and didn't have any problems, so we're apparently doing different things. I will point out that the "Recent Projects" menu item, and the two recent project buttons on the initial screen, do not update when files move around in the filesystem. So if you rename or relocate a project, you will need to File > Open it rather than using the quick-load button.
>>
>> This can be a little confusing because the buttons on the main screen only show the filename, so if you've opened something in two different places it's not immediately obvious which one is which. The Recent Projects menu does show full paths, so check and see if that matches up.
>>
>> (It now occurs to me that the existing UI is dumb, because it's showing you a quick-load button for a project that isn't there. I'll fix that.)
>>
>> The only requirement on filenames should be that the .dis65 file and the data file live in the same directory, and have the same name (just without ".dis65" for the data file). If that's not the case you'll get the opportunity to locate the file manually. Otherwise you should be able to call it anything Windows will tolerate.
>
> No, that's not it. I am talking about double-clicking the *.dis65 files to start a fresh instance of SourceGen (e.g., with my reg-edits {submitted above}) [with no other instances of SourceGen running]. It must have something to do with the "+" in the folder name, the double-dots in the filenames, long filenames, or long pathnames (my path/filename is pretty long). It is probably MicroSoft-Windows' fault!

Don't worry about it. Replacing the "+" with " plus" in the folder name has fixed the problem.
...And, matching the source and disassembly filenames..

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 20:28:06 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

6502bench SourceGen Quirks & Replies [Edited]:

> On Sunday, October 6, 2019, Andy McFadden wrote:
>> On Sunday, October 6, 2019, James Davis wrote:

>> In the disassembly window, at the bottom, when the slider bar is
>> not all the way to the left, and/or any amount or all the way to
>> the right, double-clicking on a column (e.g., the label-column*)
>> brings up the action-window for the column just to the right of
>> the column double-clicked (e.g., the attribute-column*,
>> respectively);

> That's a bug. WPF (Windows Presentation Foundation) is based on
> the philosophy that hard things should be possible, and easy things
> should be really hard. One manifestation of this is the absence of
> "what row and column was clicked on" methods in ListView and
> DataGrid. My implementation failed to take the horizontal scroll
> position into account.

> Fixed.

Thanks.

>> In the disassembly window, I keep expecting the (long or short)
>> comments and notes to work like in a word-processor (e.g.,
>> NotePad, MetaPad or {my preferred} WordPad)

> I have a "to do" list item for in-place editing of labels and
> comments. There might be a way to convince WPF to do it for me, but
> it'll likely require using a fully custom control.

> The main disassembly window is a WPF ListView control with some
> style changes, mostly to have text that extends across multiple
> columns. This was sufficiently difficult that I created a separate
> github project just to play with it
> (https://github.com/fadden/DisasmUiTest). I'm not yet experienced
> with WPF to know how hard in-place editing would be.

>> A quick double-click should not be necessary for editing
>> existing (long or short) comments and notes; two single-clicks
>> should also work within the same field {of lines--for "multi-
>> line" comments and notes} (no matter where the mouse/cursor has
>> been between single-clicks).

> I think that would make sense for in-place editing. I'm picturing
> the way Windows lets you rename a file by selecting it, and then

> doing a second single click. I don't think it feels right when the
> action is to open an editor window though.

I think it would feel better than not having anything happen at all!
Something is better than nothing. Opening an editor window is better
than being surprised when nothing at all happens; especially when one
is mentally elsewhere, ahead of the action, waiting for it to happen,
so ready to start editing the line(s) that one "can't hardly wait!"
[:-)] And, it should be easier to implement than in-place editing.

>> There needs to be a way [using Windows' standard (large) Color
>> selector dialog] to set some or all the colors used in the
>> application to user preferred colors (e.g., all the different
>> highlighting colors, window background color, etc.) vs. the
>> Users' windows-theme or Windows' standard colors; one or the
>> other of which should be selectable as a base to start with (or
>> to build upon).

> I have a long-standing request for a "dark" theme. Seems like it
> should be straightforward in WPF, but I haven't figured it out yet.

>> [For example, I would prefer all the 'SELECTED' field-
>> highlighting to be the same (e.g., dark background with light
>> lettering {e.g., blue/white}), not the (light background and
>> dark lettering) colors you have enabled for long-comments and
>> notes.--{No offense meant!}]

> That's another bug. The custom style that allows me to have long
> comments extend across multiple columns also requires me to define
> how highlighting works. In Win10, the system-wide theme has
> essentially no effect on the app, so my selected colors look the
> same as everything else in the list. I tried it just now on Win7,
> and the "high contrast" system theme significantly altered all of
> the colors in SourceGen, except for the ones in the replacement
> style. Ugh.

> So I need to figure out how to get the appropriate color out of
> the system style. There's some chance that the Win7 Aero stuff
> behaves in a totally different way from what Win10 uses, which
> would make this even more fun.

Win7 themes are kept in the "%UserProfile%\AppData\Local\Microsoft
\Windows\Themes\" folder (e.g., "Custom.theme" and others). Which
theme is currently in force is in the Windows Registry at key/value:
" HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\ Themes "/
"CurrentTheme" and the data is a drive:\path\filename. You probably

need to use the "[VisualStyles]" data section of the current theme
file.

>> Allow fonts (for individual words and characters) to be normal,
>> bold, italic, underlined, struck-through, and subscripted or
>> superscripted; like in a word-processor (e.g., WordPad {RTF});
>> especially in comments and notes.

> This is doable, but I'm a little hesitant because one of the
> primary goals of SourceGen is to generate plain text source code,
> which means either stripping the formatting (which could lose
> information, e.g., the fact that text was struck out) or generating
> comments with RTF format strings (which is not especially easy for
> humans to read). It's useful for HTML output, but I'd rather have
> people express themselves within the confines of a fixed-width
> Unicode font so that things look the same everywhere.

Forget it. It is a bad idea.

>> In the disassembly window, at the top, the column headers should
>> also be re-adjustable using the standard Windows ctrl-+, shift-
>> +, and non-standard ctrl-shift-+ key-combinations

> I'm unclear as to how standard those are in terms of column
> widths. For example, in Chrome, those change the page "zoom" level.

I got used to this in Windows Explorer and in most of the NirSoft
Utilities that show detailed listings of data. The non-standard ctrl-
shift-+ key-combination is my idea exclusively for SourceGen to reset
column widths to the basic standard it starts with or what is set in
the SourceGen settings. [I do not use any Chromium based products.
I like Microsoft's Internet Explorer and Windows Live Mail.]

>> It would be nice to be able to insert extra blank-lines, long-
>> comments and notes, above and/or below other lines and at the
>> very end, in the disassemblies.
>>
>> And/or, it would be nice to be able to drag-&-drop blank-lines,
>> long-comments and notes, up or down in the disassemblies.

> This is doable. The reason it's not done now is that everything is
> associated with a file offset, and you only get one of each thing.

> So if you add a long comment, that comment is associated with a
> specific byte in the file, and is always displayed above that byte.

> The mechanism could be made more general, e.g., multiple comments
> per offset, which can be placed above or below. Blank lines can be
> fiddled with. I think drag & drop or some other affordance would be
> necessary, since tweaking with the cosmetic aspects is awkward if
> you have to do it through a dialog window.

>> A line space is achievable as a single space in a long-comment
>> or note, but they have inline artifacts!

> It would be easy to treat an empty single-line long comment as an
> explicit blank line. There's a potential issue in that it's hard to
> tell it apart from the automatically generated blank lines. (I
could
> have a "don't put leading semicolons on blank lines" checkbox.)

>> To put anything below the last line, one would have to pad the
>> original binary/hexadecimal machine code with an extra zero
>> byte, and remove it after editing the disassembly. Would the
>> edit remain or disappear? I will have to test this.

> The .dis65 file includes the file length and CRC, so you'd have to
> update those values before SourceGen would accept the altered file.
> It would then notice that you have something with an invalid offset,
> and ignore it. So... don't do this.

Why don't you add (a choice to have) an extra line at the end of the disassembly listing that has the mnemonic "END" and all the rest of the editable fields that code lines have. Then one could edit its short-comment or put a long-comment or a note above it, if so desired. It would not need an address, nor some other things, so you could leave those fields blank or filed with stars. For example:

```
   ************************** END **************************
```

>> It would be nice to be able to load an ASCII text of another
>> disassembly created by other means and have SourceGen be able to
>> use it (detecting which assembler format to use, too), with or
>> without a binary/hexadecimal machine code file to go with it.

> This would be best done with a tool that generated a .dis65 file.
> The goal of such a thing would be to facilitate conversion of source
> code from one assembler to another by using SourceGen as a least-
> common-denominator format. The trouble is that SourceGen is far

less
> expressive than a typical assembler, and you'd lose a lot. But if
> you can keep most of the labels and comments, that'd be an easier
> place to start from.

Okay, that should be your second 6502bench tool: a translator that takes a pre-existing ASCII/ANSI text file, a disassembly listing (with or without the address, code bytes, line numbers, and other data that preceedes the label field), and creates a .dis65 file from it. (It should also work with SourceGen's text-file exports.) [But, can it be done without a binary-source file? The tool might also have to generate one for SorceGen to work with; which means the tool would also be an assembler (e.g., "The Fadden Assembler").]

Your third 6502bench tool should do something similar for .sym65; translating from pre-existing ASCII/ANSI text file, symbol tables (e.g., AppleWin {1.11.2.1}'s Apple2e.sym) and/or disassembly listings (e.g., their "Equate Tables"). [Should be a part of "The Fadden Assembler"!]

[Call the whole thing "The Fadden Editor/Assembler/Disassembler (FEDASM65)"; or keep it as "6502bench" if you like.]

[Then I could re-create all my "Apple II Memory Equates for Assembly Programmers" text-edits in SourceGen. I have added quite a lot to some of the published Apple II disassembly listing I have. I have been re-analyzing them off and on for the last couple of years. They are verbosely commented; especially the 'Equates' documents; with quite a few new (undocumented) entry points (for m/l programming {tricks}) I have discovered. They are all works in progress.]

>> The settings keep resetting/reverting to 64tass on the last
>> three tabs. They do not stay set to Merlin from session to
>> session.

> This isn't a bug so much as it's just bad UI design. It annoys me
> too.

> If you launch SourceGen v1.0, the "Quick Set" box has three
> buttons, "Default", "cc65", and "Merlin 32". You click the button,
> and the various fields get reconfigured. Simple.

> In SourceGen v1.1, I added support for 64tass, and realized that
> my previous scheme didn't scale well, and made it difficult to add
> new assemblers. So instead of a pair of buttons, you have a pop-up
> menu, with a "Set" button to actually apply the values.

> The problem, as you have noted, is that it looks like something is
> configured for 64tass, even though in practice nothing happens when
> you select a different assembler from the pop-up menu.

> So the question is how to change the UI to make sense. One
> possibility is to get rid of "Set" and just make the combo box live:
> selecting a different assembler immediately updates all values. A
> new entry, "Custom", would hold the state of any recent edits. The
> pop-up would either show the name of the assembler (if all fields
> match) or "custom" (if any don't). So as soon as you make a change
> to any field, it switches to "custom" and obliterates any previously
> established custom setting... but I don't see a way around doing so
> that doesn't require wrapping a heavyweight customization UI around
> something that should be simple.

I see in my "C:\Program Files\6502bench\SourceGen-settings" (no extension) file, that it only has Merlin listed:

"fmt-expression-mode":"Merlin",
"asm-config-Merlin32":"{\"ExecutablePath\":\"\",\"ColumnWidths\ ":
[11,4,15,50]}",
"srcgen-default-asm":"Merlin32",

And, all my disassemblies are in Merlin format, so "64tass" is just an annoyance.

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

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 08 Oct 2019 21:01:34 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.4-dev1 is now available. Key features:

- Platform and project symbols now have explicit widths.
- Extension script formatting capabilities have been expanded.
- ProDOS 8 symbol file and script have been enhanced.

Platform and project symbols are used to define labels for addresses outside the project file, like zero-page locations, jump vectors, and ROM entry points. Being able to identify pointers, vectors, and multi-byte tables makes the auto-disassembly more accurate when something references PTR+1 or BUFFER+23.

After adding widths to the symbol definitions, I was able to remove the formatting from about 110

locations in the Applesoft disassembly, mostly for access to floating-point storage locations like FAC (6 bytes). This also allowed me to fully describe the ProDOS 8 MLI globals page, so those definitions have been added to the symbol file that ships with SourceGen.

This may require updates to existing projects with project symbols, as the external symbols were effectively being treated as 3 bytes wide in v1.3. So some things that used to pick up "PTR+1" won't anymore until you define PTR as 2 bytes.

The set of inline data format options available to extension scripts has been expanded to include strings. Also, scripts can now get at project and user symbols (i.e. labels on things inside the file), and can query the data structure that defines the mapping of offsets to addresses. The benefits of these changes can be demonstrated with an example.

Consider Glen Bredon's "Cat Doctor" utility. Given this P8 Open call:

```
2154: 20 00 bf          jsr     P8_MLI
2157: c8              .dd1    P8_OPEN
2158: bf 21           .dd2    L21BF
```

You could double-click on the .dd2 to jump to the data area, which looks like this:

```
21bf: 03        L21BF   .dd1    $03
21c0: cf              .dd1    $cf
21c1: 21              .dd1    $21
21c2: 00              .dd1    $00
21c3: b0              .dd1    $b0
21c4: 00        L21C4   .dd1    $00
```

The updated ProDOS 8 script converts $21bf to a file offset, and formats the data at that location automatically. It now looks like this:

```
21bf: 03        L21BF   .dd1    3
21c0: cf 21           .dd2    L21CF
21c2: 00 b0           .dd2    $b000
21c4: 00        L21C4   .dd1    0
```

Of particular note is the second line, which is now correctly recognized as a pointer to a pathname buffer, and has a label generated automatically.

This is handy, but the real trick when poking around in Cat Doctor is the inline string function. Calls to a certain address are followed by string data, using the same sort of mechanism the ProDOS MLI does to skip past the function code and param block pointer.

With the new formatting capabilities, if you see a JSR to the string-print code, you can format the block of data that follows the JSR as an inline null-terminated string, and the disassembler will step over it. Hard-coding locations into the extension script is annoying, so we use the label lookup feature to avoid doing so.

Given this:

```
0872: 20 8c 22          jsr     $228c
0875: 21 c3             and     ($c3,x)
0877: c1 d4             cmp     ($d4,x)
0879: a0 c4             ldy     #$c4
087b: cf c3 d4 cf+      .str    “OCTOR 6.8”,$8d
0885: 19                .dd1    $19
0886: c3 ef f0 f9+      .str    “Copyright 1990  by Glen Bredon”,$8d,$8d
08a6: 00                .dd1    $00
08a7: 84                .dd1    $84
08a8: da                .dd1    $da
```

If you add an appropriate script (such as SourceGen/Examples/Scripts/InlineNullTerm.cs), and set the label at $228c to "PrintInlineZString", SourceGen will disassemble this to:

```
0872: 20 8c 22          jsr     PrintInlineZString
0875: 21 c3 c1 d4+      .zstr   $21,“CAT DOCTOR 6.8”,$8d,$19,“Copyright 1990  by
Glen Bredon”
                  +     $8d,$8d
08a7: 84 da             sty     $da
```

(You may have to hit F5 to refresh after editing the label. This is due to SourceGen trying to be efficient and only do a partial re-analysis after a label change. I'm still trying to figure out a good way to handle this.)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 19 Oct 2019 18:28:09 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.4-dev2 is now available. Changes:

- Added .junk/.align directives. These allow you to mark the contents of a memory region as irrelevant, and indicate whether you think it exists to align the output address to a page boundary or other power-of-two value.
- Added I/O-direction-specific symbol declarations. These are for memory-mapped I/O locations that behave differently when read or written. For example, reading from $C000 gets you the last character read, while writing to $C000 changes the state of the 80-column hardware.
- Added address mirroring. This is mostly for systems like the Atari 2600, where RAM, ROM, and I/O appear at multiple addresses.
- Restored the ability to treat BRK as a two-byte instruction (choose 1-byte or 2-byte from the project properties).
- Added "find previous" (Shift+F3).
- Changed the way extension scripts interact with the symbol table, so that we can update the

display when a symbol that a plugin cares about is updated.
 - Updated the style on the main code list so it looks the same on all lines, regardless of the system theme. Added a switch to enable "dark mode".
 - Changed the "quick set" combo box in the last couple of app settings tabs to set the assembler values immediately. (The previous UI was misleading..)
 - Changed the sort order on the equate list so all constants come before all addresses.
 - Improved navigation back to a local variable table entry. Before, if you selected a local variable, double-clicked on a reference, and then hit "back", the selection would be on the following instruction rather than the variable table entry.

Source code and pre-built Windows binaries are available from
https://github.com/fadden/6502bench/releases

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 20 Oct 2019 05:47:40 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

Still using v1.3.2

Is there a way to make all symbols/labels global and exported, easily, all at once?

I have enhanced several of the sym65 files via text editor, but is there a way to tell SourceGen to make the symbols global and exported from the sym65 file?

Every once in a while I come across the means to do it for individual labels in SourceGen, but most of the time I can never find the property page that allows setting a symbol/label to global and exported. Where is it?

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 20 Oct 2019 14:51:05 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Saturday, October 19, 2019 at 10:47:42 PM UTC-7, James Davis wrote:
> Is there a way to make all symbols/labels global and exported, easily, all at once?
>
> I have enhanced several of the sym65 files via text editor, but is there a way to tell SourceGen to make the symbols global and exported from the sym65 file?

The idea behind an exported symbol is that it's available to import into other projects. All symbols in a .sym65 file are global and exported: external address symbols are inherently global, and you can import them into any project by including the file.

The symbol files can live in the RuntimeData directory for system-level stuff, or in the project directory, so you can create your own versions and keep them with the project file. The order in which the files are included is significant, meaning you can replace the "built in" definitions with your own and distribute the changes with your project. (You can see an example in Examples/A2-Amper-fdraw, which has a .sym65 for the fdraw library entry points.)

> Every once in a while I come across the means to do it for individual labels in SourceGen, but most of the time I can never find the property page that allows setting a symbol/label to global and exported. Where is it?

You can set it when editing an address label (double-click on a label).

Currently the "export" flag has two uses: (1) exported symbols show up in the HTML dump; (2) you can import one project's external symbols into another project as a crude way to do multi-binary disassembly. Are you doing one of these things, or something else? (If it's #1, it would be far easier to add an "include all global symbols". Then you could just turn off the label localizer to include everything.)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 20 Oct 2019 19:36:24 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Sunday, October 20, 2019 at 7:51:07 AM UTC-7, fadden wrote:
> On Saturday, October 19, 2019 at 10:47:42 PM UTC-7, James Davis wrote:
>> Is there a way to make all symbols/labels global and exported, easily, all at once?
>>
>> I have enhanced several of the sym65 files via text editor, but is there a way to tell SourceGen to make the symbols global and exported from the sym65 file?
>
> The idea behind an exported symbol is that it's available to import into other projects. All symbols in a .sym65 file are global and exported: external address symbols are inherently global, and you can import them into any project by including the file.
>

How do I include the file?

When I try to import my "APPLE2.ROM.dis65"* (Apple II Plus ROM Disassembly) into another (new) project, it says there are no exported symbols [except for the few individual symbols that I have since {miraculously** (see below)} made 'global and exported' from within SourceGen].

[* It uses my {highly commented} "Apple2Plus_UC.sym65" (Apple II Plus ROM Symbols) file that is in the project folder.]

> The symbol files can live in the RuntimeData directory for system-level stuff, or in the project

directory, so you can create your own versions and keep them with the project file. The order in which the files are included is significant, meaning you can replace the "built in" definitions with your own and distribute the changes with your project. (You can see an example in Examples/A2-Amper-fdraw, which has a .sym65 for the fdraw library entry points.)
>
>> Every once in a while I come across the means to do it for individual labels in SourceGen, but most of the time I can never find the property page that allows setting a symbol/label to global and exported. Where is it?
>
> You can set it when editing an address label (double-click on a label).
>

This option is rarely there/available in my version of SourceGen! [** I think it is a miracle when it shows up!]

> Currently the "export" flag has two uses: (1) exported symbols show up in the HTML dump; (2) you can import one project's external symbols into another project as a crude way to do multi-binary disassembly. Are you doing one of these things, or something else? (If it's #1, it would be far easier to add an "include all global symbols". Then you could just turn off the label localizer to include everything.)

Number (2); see replies above.

What do you mean by, "multi-binary disassembly"?

How do I 'add an "include all global symbols"'? Where and to what?
[I think you mean: During setup for HTML/text output of a listing (?).]

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 20 Oct 2019 23:43:56 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Sunday, October 20, 2019 at 12:36:26 PM UTC-7, James Davis wrote:
>> The idea behind an exported symbol is that it's available to import into other projects. All symbols in a .sym65 file are global and exported: external address symbols are inherently global, and you can import them into any project by including the file.
>
> How do I include the file?

You include a .sym65 file via Edit > Project Properties, Symbol Files tab, then "Add".

You import global symbols from a .dis65 file via Edit > Project Properties, Project Symbols tab, then "Import".

But I think you know that already. :-)

>> You can set it when editing an address label (double-click on a label).
>
> This option is rarely there/available in my version of SourceGen! [** I think it is a miracle when it shows up!]

I will amend my previous statement: double-click on a label *on code or data that is inside the file*. The window title will be "Edit Label".

If you double-click on a label that is part of a .EQ (equate) directive, it will either open the project property editor (title is "Edit Symbol"), or do nothing (can't edit .sym65 files from here). Double-clicking on the label of a .VAR (local variable) directive won't work either.

The point of "export" is to export symbols associated with the contents of the file being disassembled. Exporting lists of external addresses that were imported from somewhere else isn't a feature.

>> Currently the "export" flag has two uses: (1) exported symbols show up in the HTML dump; (2) you can import one project's external symbols into another project as a crude way to do multi-binary disassembly. [...]
>
> Number (2); see replies above.
>
> What do you mean by, "multi-binary disassembly"?

An arrangement where there are multiple binaries that interact with each other. As opposed to having a single large binary that encompasses everything.

If you've got one file fully disassembled before you start on the next, it works fine. If you're ping-ponging between two files, the manual import step is annoying.

What I'm trying to accomplish with "export" is similar to Merlin 32 ENT/EXT directives. Merlin's ENT directive adds as ENTry for the symbol to a table included in a linkable object file. Other object files declare the symbol as EXTternal, and it gets resolved at link time. Marking a symbol "export" in SourceGen would cause an ENT directive to be emitted in generated sources if SourceGen had more thorough multi-binary support.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Fri, 25 Oct 2019 01:46:16 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Sunday, October 20, 2019 at 4:43:58 PM UTC-7, fadden wrote:
> On Sunday, October 20, 2019 at 12:36:26 PM UTC-7, James Davis wrote:
>>> The idea behind an exported symbol is that it's available to import into other projects. All symbols in a .sym65 file are global and exported: external address symbols are inherently global,

and you can import them into any project by including the file.
>>
>> How do I include the file?
>
> You include a .sym65 file via Edit > Project Properties, Symbol Files tab, then "Add".
>
> You import global symbols from a .dis65 file via Edit > Project Properties, Project Symbols tab, then "Import".
>
> But I think you know that already. :-)
>
>>> You can set it when editing an address label (double-click on a label).
>>
>> This option is rarely there/available in my version of SourceGen! [** I think it is a miracle when it shows up!]
>
> I will amend my previous statement: double-click on a label *on code or data that is inside the file*. The window title will be "Edit Label".
>
> If you double-click on a label that is part of a .EQ (equate) directive, it will either open the project property editor (title is "Edit Symbol"), or do nothing (can't edit .sym65 files from here). Double-clicking on the label of a .VAR (local variable) directive won't work either.
>
> The point of "export" is to export symbols associated with the contents of the file being disassembled. Exporting lists of external addresses that were imported from somewhere else isn't a feature.
>
>>> Currently the "export" flag has two uses: (1) exported symbols show up in the HTML dump; (2) you can import one project's external symbols into another project as a crude way to do multi-binary disassembly. [...]
>>
>> Number (2); see replies above.
>>
>> What do you mean by, "multi-binary disassembly"?
>
> An arrangement where there are multiple binaries that interact with each other. As opposed to having a single large binary that encompasses everything.
>
> If you've got one file fully disassembled before you start on the next, it works fine. If you're ping-ponging between two files, the manual import step is annoying.
>
> What I'm trying to accomplish with "export" is similar to Merlin 32 ENT/EXT directives. Merlin's ENT directive adds as ENTry for the symbol to a table included in a linkable object file. Other object files declare the symbol as EXTternal, and it gets resolved at link time. Marking a symbol "export" in SourceGen would cause an ENT directive to be emitted in generated sources if SourceGen had more thorough multi-binary support.

OK, I think I understand it better now.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 26 Oct 2019 01:32:14 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.4-alpha1 is now available. Changes:

 - Added a message list to point out subtle problems, like bad formatting or references to nonexistent symbols.
 - Added an instruction chart. The window displays a summary of all 256 instructions for the selected CPU (6502, 65C02, or 65816).
 - Tweaked some stuff in the UI.
 - Updated the tutorial with a section on inline data formatting, and a section on extension scripts.

Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 26 Oct 2019 01:38:41 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

I have updated the projects at https://bigflake.com/disasm/ to use the features available in SourceGen v1.4, and added a new project.

Bob Bishop's Micro-Painter fascinated me because of two features. First, it had a flood fill that actually looked like a flood. Second, it had a "zoom" mode that used the lo-res screen to let you edit the hi-res screen.

If you haven't seen the program in action before, I captured a brief video here: https://www.youtube.com/watch?v=ylpZI_E1T88

I always wondered how it worked. Now I know. :-)

  https://bigflake.com/disasm/a2-micropainter/

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 26 Oct 2019 04:54:31 GMT
View Forum Message <> Reply to Message

Originally posted by: Frank M.

I've been playing around with Apple-1 stuff and came up with a symbol list for the machine, if you're interested in adding it to SourceGen.

f

Apple-1 Symbol List.sym65

```
*Apple-1 symbol list for SourceGen
*2019 Frank Milliron


*System Symbols

XAML            @ $24        ;Last "opened" location Low
XAMH            @ $25        ;Last "opened" location High
STL             @ $26        ;Store address Low
STH             @ $27        ;Store address High
L               @ $28        ;Hex value parsing Low
H               @ $29        ;Hex value parsing High
YSAV            @ $2A        ;Used to see if hex value is given
MODE            @ $2B        ;$00=XAM, $7F=STOR, $AE=BLOCK XAM

IN              @ $0200,$027F ;Input buffer

USER4           @ $0280,0FFF  ;User RAM (4k system)
USER8           @ $0280,1FFF  ;User RAM (8k system)

KBD             @ $D010      ;PIA.A keyboard input
KBDCR           @ $D011      ;PIA.A keyboard control register
DSP             @ $D012      ;PIA.B display output register
DSPCR           @ $D013      ;PIA.B display control register



*WOZACI (Woz's Apple Cassette Interface) Symbols

HEX1L           @ $24        ;End address of dump block
HEX1H           @ $25
HEX2L           @ $26        ;Begin address of dump block
HEX2H           @ $27
SAVEINDEX       @ $28        ;Save index in input buffer
LASTSTATE       @ $29        ;Last input state

FLIP            @ $C000      ;Output flip-flop
TAPEIN          @ $C081      ;Tape input

WOZACI          @ $C100      ;Apple Cassette Interface Entry
NEXTCHAR        @ $C10C
KBDWAIT         @ $C10D
```

```
NEXTCMD         @ $C125      ;Start parsing first or a new tape command
NEXTCHR         @ $C12F
DIG           @ $C153
HEXSHIFT        @ $C159
GOESC           @ $C163      ;Return to monitor (ESCAPE), prints \ first
SEP           @ $C166      ;Separating period found. Copy HEX1 to HEX2
WRITE           @ $C170      ;Write a block of memory to tape
WRNEXT          @ $C175
WBITLOOP        @ $C17C
RESTIDX         @ $C189
READ          @ $C18D      ;Read from tape
NOTSTART        @ $C198
RDBYTE          @ $C1A4
RDBIT           @ $C1A6
FULLCYCLE       @ $C1BC
CMPLEVEL        @ $C1BF
WHEADER         @ $C1CC      ;Write header to tape
HCOUNT          @ $C1CE
WRITEBIT        @ $C1DB      ;Write a full bit cycle
WDELAY          @ $C1E0
WDELAY0         @ $C1E7
WRITE1          @ $C1EA
INCADDR         @ $C1F1      ;Increment current address and compare with last
NOCARRY         @ $C1FF      ;RTS
```

*Apple-1 Cassette BASIC

```
BASIC           @ $E000      ;Cold start location
WARMSTART       @ $E2B3      ;Warm start location
```

*Apple-1 ROM

```
RESET           @ $FF00      ;ROM cold start location
NOTCR           @ $FF0F
ESCAPE          @ $FF1A      ;Escape back to monitor
GETLINE         @ $FF1F
BACKSPACE       @ $FF26
NEXTCHAR        @ $FF29
SETSTOR         @ $FF40
SETMODE         @ $FF41
BLSKIP          @ $FF43
NEXTITEM        @ $FF44
NEXTHEX         @ $FF5F      ;trying to parse a new hex value
DIG           @ $FF6E
```

```
HEXSHIFT        @ $FF74
NOTHEX          @ $FF7F
TONEXTITEM      @ $FF91
RUN             @ $FF94      ;RUN user's program from last opened location
NOTSTOR         @ $FF97      ;We're not in Store mode
SETADR          @ $FF9B
NXTPRNT         @ $FFA4      ;Print address and data from this address
PRDATA          @ $FFBA
XAMNEXT         @ $FFC4
MOD8CHK         @ $FFD6
PRBYTE          @ $FFDC      ;Subroutine to print a byte in A in hex form (destructive)
PRHEX           @ $FFE5      ;Subroutine to print a hexadecimal digit
ECHO            @ $FFEF      ;Subroutine to print a character to the terminal
NMI_VEC         @ $FFFA      ;NMI vector   ($0F00)
RESET_VEC       @ $FFFC      ;RESET vector ($FF00)
IRQ_VEC         @ $FFFE      ;IRQ vector   ($0000)
```

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 26 Oct 2019 04:57:59 GMT
View Forum Message <> Reply to Message

Originally posted by: Frank M.

I also have a much-expanded symbol list for the Apple /// that's in a semi-not finished state currently.
f

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 26 Oct 2019 15:43:27 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Friday, October 25, 2019 at 9:54:33 PM UTC-7, Frank M. wrote:
> I've been playing around with Apple-1 stuff and came up with a symbol list for the machine, if you're interested in adding it to SourceGen.

Nice! I've added a .sym65 for it and updated the system definition list:

https://github.com/fadden/6502bench/commit/6344ea59bb1429230 521b0056071b76177ec1d70

Please give it a once-over. You can apparently leave comments directly on that change list by clicking on lines, though you might need to be signed in to do so.

I made a couple of changes:

> IN @ $0200,$027F ;Input buffer

Multi-byte items are "<address> <length>" rather than "<start>,<end>", so this became "$0200 128".

> USER4 @ $0280,0FFF ;User RAM (4k system)
> USER8 @ $0280,1FFF ;User RAM (8k system)

These would cause any memory reference outside the bounds of the program being disassembled to appear as "USER4 + <offset>", rather than just "$0F12" or whatever. That seemed undesirable, so I changed it to:

USER = $0280

So there's a constant for the start but it doesn't replace all unknown addresses.

> XAML @ $24 ;Last "opened" location Low
...
> HEX1L @ $24 ;End address of dump block

FYI: if you have multiple symbols for the same address in the same platform symbol file, the "lookup by address" function will resolve it alphabetically. So "LDA $24" will be auto-formatted as "LDA HEX1L" because it's lexically before "XAML". (You can of course set the LDA operand to the symbol of your choice, but that's a manual step.)

> I also have a much-expanded symbol list for the Apple /// that's in a semi-not finished state currently.

Another machine about which I know nearly nothing. :-) Anything that helps the Apple ///-ers would be welcome.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 26 Oct 2019 19:01:27 GMT
View Forum Message <> Reply to Message

Originally posted by: Frank M.

On Saturday, October 26, 2019 at 8:43:29 AM UTC-7, fadden wrote:
> On Friday, October 25, 2019 at 9:54:33 PM UTC-7, Frank M. wrote:
>> I've been playing around with Apple-1 stuff and came up with a symbol list for the machine, if you're interested in adding it to SourceGen.
>
> Nice! I've added a .sym65 for it and updated the system definition list:
>
> https://github.com/fadden/6502bench/commit/6344ea59bb1429230521b0056071b76177ec1d70

>
> Please give it a once-over. You can apparently leave comments directly on that change list by clicking on lines, though you might need to be signed in to do so.
>
> I made a couple of changes:
>
>> IN @ $0200,$027F ;Input buffer
>
> Multi-byte items are "<address> <length>" rather than "<start>,<end>", so this became "$0200 128".
>
>> USER4 @ $0280,0FFF ;User RAM (4k system)
>> USER8 @ $0280,1FFF ;User RAM (8k system)
>
> These would cause any memory reference outside the bounds of the program being disassembled to appear as "USER4 + <offset>", rather than just "$0F12" or whatever. That seemed undesirable, so I changed it to:
>
> USER = $0280
>
> So there's a constant for the start but it doesn't replace all unknown addresses.
>
>> XAML @ $24 ;Last "opened" location Low
> ...
>> HEX1L @ $24 ;End address of dump block
>
> FYI: if you have multiple symbols for the same address in the same platform symbol file, the "lookup by address" function will resolve it alphabetically. So "LDA $24" will be auto-formatted as "LDA HEX1L" because it's lexically before "XAML". (You can of course set the LDA operand to the symbol of your choice, but that's a manual step.)
>
>> I also have a much-expanded symbol list for the Apple /// that's in a semi-not finished state currently.
>
> Another machine about which I know nearly nothing. :-) Anything that helps the Apple ///-ers would be welcome.

I made up the labels USER4/8 anyway. :)

Yes, the Apple /// is a bit under-documented. I've been using MAME memory dumps and SourceGen to look at various programs and found a lot of references to things not in the available memory maps.

The Apple-1 is pretty well documented. Though it's such a limited machine there's not a lot to get. Side note, the only emulator which seems to work properly for this machine is OpenEmulator. For

Apple /// it's MAME.

http://openemulator.github.io/

f

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 28 Oct 2019 21:55:37 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

I've long wondered what sort of algorithm DDDeluxe 5.0 applied when packing disk images. There were a number of competing programs in the pre-ShrinkIt days. ISTR DDDeluxe working better than most.

I started disassembling it, but kept finding lots of weird self-modifying code. Not the usual sorts of "set up the pointer I'm about to use a lot", but rather "let's copy this 3-byte JMP instruction on top of other code".

I wasn't sure why it was doing those things, until I found this:

..str ‘This message is for you, the asshole who tryto disassemble my ’
+ ‘program D.D.DeLUXE V5.0:I know why you do this: You don't hav’
+ ‘e enoughimagination to program your own packing algorithmand p’
+ ‘rogram.Because you are not enough intelligent, you preferto co’
+ ‘py what I did... That's the true!Don't forget this program is’
+ ‘ a SHAREWARE and it isa copyrighted product by LOGIX INNOVATIO’
+ ‘NS 1989.You can be pursuit for $10000 if you copy anypart of t’
+ ‘his program without the autorization ofLOGIX INNOVATIONS. Fuc’
+ ‘k you, NO-BRAND !’

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Hugh Hood on Mon, 28 Oct 2019 23:14:24 GMT
View Forum Message <> Reply to Message

in article d9d34128-0bbb-49f0-a7b1-47e1dfaf7f75@googlegroups.com, fadden at thefadden@gmail.com wrote on 10/28/19 4:55 PM:
>
> I wasn't sure why it was doing those things, until I found this:
>

>

Andy,

That's hilarious!

Thanks for putting a big grin on my face today.

Hugh Hood

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 29 Oct 2019 05:14:14 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

I was disassembling an old game and thinking, "meh". The Budge 3D stuff had some nifty bits, but this other one just sort of sprawled in not very inspiring ways.

So I took a break and disassembled the $C600 Disk][firmware.

Genius.

https://bigflake.com/disasm/a2-boot/

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 29 Oct 2019 08:37:53 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Monday, October 28, 2019 at 10:14:16 PM UTC-7, fadden wrote:
> I was disassembling an old game and thinking, "meh". The Budge 3D stuff had some nifty bits, but this other one just sort of sprawled in not very inspiring ways.
>
> So I took a break and disassembled the $C600 Disk][firmware.
>
> Genius.
>
> https://bigflake.com/disasm/a2-boot/

Downloaded the following:

Apple II 5.25 Disk Boot Disassembly

Applesoft Disassembly
Autostart ROM Disassembly

What is the assembler (style) you are using for the disassembly listings?
Could you add listings in the Merlin 32 format? It is closer to EDASM, which Apple II had, back in the day.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 29 Oct 2019 09:44:24 GMT
View Forum Message <> Reply to Message

Originally posted by: Anthony Adverse

On Tuesday, October 29, 2019 at 8:55:39 AM UTC+11, fadden wrote:
> I've long wondered what sort of algorithm DDDeluxe 5.0 applied when packing disk images. There were a number of competing programs in the pre-ShrinkIt days. ISTR DDDeluxe working better than most.
>
> I started disassembling it, but kept finding lots of weird self-modifying code. Not the usual sorts of "set up the pointer I'm about to use a lot", but rather "let's copy this 3-byte JMP instruction on top of other code".
>
> I wasn't sure why it was doing those things, until I found this:
>
> .str ‘This message is for you, the asshole who tryto disassemble my ’
> + ‘program D.D.DeLUXE V5.0:I know why you do this: You don't hav’
> + ‘e enoughimagination to program your own packing algorithmand p’
> + ‘rogram.Because you are not enough intelligent, you preferto co’
> + ‘py what I did... That's the true!Don't forget this program is’
> + ‘ a SHAREWARE and it isa copyrighted product by LOGIX INNOVATIO’
> + ‘NS 1989.You can be pursuit for $10000 if you copy anypart of t’
> + ‘his program without the autorization ofLOGIX INNOVATIONS. Fuc’
> + ‘k you, NO-BRAND !’

I thought there was some level of consensus that DDD used huffman encoding?

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 29 Oct 2019 15:21:25 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Tuesday, October 29, 2019 at 1:37:56 AM UTC-7, James Davis wrote:
> Downloaded the following:
>

> Apple II 5.25 Disk Boot Disassembly
> Applesoft Disassembly
> Autostart ROM Disassembly
>
> What is the assembler (style) you are using for the disassembly listings?
> Could you add listings in the Merlin 32 format? It is closer to EDASM, which Apple II had, back in the day.

For the HTML output, I'm using Merlin-style string delimiters (but with curly quotes), default pseudo-ops, letter case, and expression style, and column widths tweaked slightly for each project (e.g. 12-8-16-100).

Of course, once you have the SourceGen project and the binary (both of which are included in the download ZIPs), you can generate HTML listings or fully working assembly code in whatever format you like. You can File > Assemble, select Merlin 32, and generate a working .S file. (If you have Merlin 32 installed, it'll even run the assembler for you just to prove that the output is correct.)

SourceGen doesn't yet support "retro" assemblers directly because that imposes additional constraints, like maximum file size limits, and it's harder to regression-test because you have to bounce everything through an emulator. Merlin 32 is close enough to the original that you probably wouldn't have to fix the syntax.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 29 Oct 2019 15:33:45 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Tuesday, October 29, 2019 at 2:44:25 AM UTC-7, Anthony Adverse wrote:
> I thought there was some level of consensus that DDD used huffman encoding?

The original DDD did. Somebody did a disassembly back in the day, and I used that to add support to CiderPress (https://github.com/fadden/ciderpress/blob/master/diskimg/DDD .cpp). It's essentially RLE + static Huffman. The 20 most-frequently-appearing symbols get output as prefix codes, the rest are output as 9-bit values.

I expect DDDeluxe is also Huffman based, since pretty much everything was until LZWhatever took over. But the devil is in the details, and the details are in the disassembly.

So I can (1) fight my way through code that doesn't want to be understood, (2) reverse-engineer the format by stuffing data in and seeing what comes out, or (3) decide nobody cares about the format and leave it rotting in the dustbin of history.

Currently going with #3.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 30 Oct 2019 02:48:23 GMT
View Forum Message <> Reply to Message

Originally posted by: James Davis

On Tuesday, October 29, 2019 at 8:21:27 AM UTC-7, fadden wrote:
> On Tuesday, October 29, 2019 at 1:37:56 AM UTC-7, James Davis wrote:
>> Downloaded the following:
>>
>> Apple II 5.25 Disk Boot Disassembly
>> Applesoft Disassembly
>> Autostart ROM Disassembly
>>
>> What is the assembler (style) you are using for the disassembly listings?
>> Could you add listings in the Merlin 32 format? It is closer to EDASM, which Apple II had, back in the day.
>
> For the HTML output, I'm using Merlin-style string delimiters (but with curly quotes), default pseudo-ops, letter case, and expression style, and column widths tweaked slightly for each project (e.g. 12-8-16-100).
>
> Of course, once you have the SourceGen project and the binary (both of which are included in the download ZIPs), you can generate HTML listings or fully working assembly code in whatever format you like. You can File > Assemble, select Merlin 32, and generate a working .S file. (If you have Merlin 32 installed, it'll even run the assembler for you just to prove that the output is correct.)
>
> SourceGen doesn't yet support "retro" assemblers directly because that imposes additional constraints, like maximum file size limits, and it's harder to regression-test because you have to bounce everything through an emulator. Merlin 32 is close enough to the original that you probably wouldn't have to fix the syntax.

Sorry Andy, I figured it out before I read this.

Load the dis65 and Export using my 6502bench SourGen setup.

You are so quick with your disassemblies that rather than use 6502bench SourGen myself, I'll just hire you to do the jobs I want done. I'll send you the info if and when.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 30 Oct 2019 04:39:41 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Tuesday, October 29, 2019 at 7:48:24 PM UTC-7, James Davis wrote:

> You are so quick with your disassemblies that rather than use 6502bench SourGen myself, I'll just hire you to do the jobs I want done. I'll send you the info if and when.

My hourly rate: ONE MILLLLLION DOLLARS

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 30 Oct 2019 04:54:16 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

WAIT just a second...

The listing of the original (non-autostart) monitor ROM, as found in the Apple II Reference Manual, says:

```
fca8: 38        WAIT      sec
fca9: 48        WAIT2     pha
fcaa: e9 01     WAIT3     sbc   #$01        ;1.0204 usec
fcac: d0 fc               bne   WAIT3       ;(13+2712*A+512*A*A)
fcae: 68                  pla
fcaf: e9 01               sbc   #$01
fcb1: d0 f6               bne   WAIT2
fcb3: 60                  rts
```

So there are two comments, and both seem completely wrong.

If we consult Apple II Monitors Peeled page 85, it says the formula is:

2.5A**2 + 13.5A + 13 machine cycles of 1.023 microseconds

However, my understanding is that the 6502 is running at 1.023MHz, so each cycle actually takes 0.9775 usec. Am I missing something, or did they get the math wrong?

The old monitor ROM listing in the Apple II Reference Manual had a lot more comments than the autostart version, so I transcribed it in SourceGen:
https://bigflake.com/disasm/a2-f8rom/OrigF8ROM.html

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 30 Oct 2019 17:17:16 GMT
View Forum Message <> Reply to Message

Originally posted by: awanderin

fadden <thefadden@gmail.com> writes:

> WAIT just a second...
>
> The listing of the original (non-autostart) monitor ROM, as found in the Apple II Reference Manual, says:
>
> fca8: 38 WAIT sec
> fca9: 48 WAIT2 pha
> fcaa: e9 01 WAIT3 sbc #$01 ;1.0204 usec
> fcac: d0 fc bne WAIT3 ;(13+2712*A+512*A*A)
> fcae: 68 pla
> fcaf: e9 01 sbc #$01
> fcb1: d0 f6 bne WAIT2
> fcb3: 60 rts
>
> So there are two comments, and both seem completely wrong.
>
> If we consult Apple II Monitors Peeled page 85, it says the formula is:
>
> 2.5A**2 + 13.5A + 13 machine cycles of 1.023 microseconds
>
> However, my understanding is that the 6502 is running at 1.023MHz, so each cycle actually takes 0.9775 usec. Am I missing something, or did they get the math wrong?
>
> The old monitor ROM listing in the Apple II Reference Manual had a lot more comments than the autostart version, so I transcribed it in SourceGen:
> https://bigflake.com/disasm/a2-f8rom/OrigF8ROM.html

The CPU has 64 clock periods of 14 * (1 / 14.318181 MHz) or 0.978µs and
one stretched period of 16 * (1 / 14.318181 MHz) or 1.117µs, which gives
an average clock period of 0.980µs. That works out to an average clock
speed of 1.0205 MHz.

I think the cycle count formula is 5 * a * a + 11 * a + 13, and that
includes the JSR/RTS.

--
Jerry awanderin at gmail dot com

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 30 Oct 2019 23:41:05 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.4 is now available. Key changes since v1.3:

- External address symbols, defined in symbol files and the project, have been enhanced:
  - Symbols have widths, so you can declare pointers and buffers.
  - Symbols may be unidirectional (read or write), for memory-mapped I/O.
  - Symbols may be mirrored to multiple addresses (e.g. Atari 2600).
- Added .junk/.align directives.
- Added a message list that appears when problems are found.
- Added a CPU instruction reference chart.
- Added an option to treat BRK as two bytes.
- Extension script formatting capabilities have been expanded.
- Various UI improvements (e.g. "dark mode" for main listing, "find previous", slightly cleaner Info panel).

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 30 Oct 2019 23:45:02 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Wednesday, October 30, 2019 at 10:17:23 AM UTC-7, awanderin wrote:
>> fcaa: e9 01 WAIT3 sbc #$01 ;1.0204 usec
[...]
>> If we consult Apple II Monitors Peeled page 85, it says the formula is:
>>
>> 2.5A**2 + 13.5A + 13 machine cycles of 1.023 microseconds

> The CPU has 64 clock periods of 14 * (1 / 14.318181 MHz) or 0.978µs and
> one stretched period of 16 * (1 / 14.318181 MHz) or 1.117µs, which gives
> an average clock period of 0.980µs. That works out to an average clock
> speed of 1.0205 MHz.

Ah, that explains the number in the comment. Doesn't explain why apparently nobody at Apple understood the relationship between MHz and microseconds. :-)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Michael J. Mahon on Thu, 31 Oct 2019 07:24:49 GMT
View Forum Message <> Reply to Message

fadden <thefadden@gmail.com> wrote:
> On Wednesday, October 30, 2019 at 10:17:23 AM UTC-7, awanderin wrote:
>>> fcaa: e9 01 WAIT3 sbc #$01 ;1.0204 usec
> [...]
>>> If we consult Apple II Monitors Peeled page 85, it says the formula is:

>>>
>>> 2.5A**2 + 13.5A + 13 machine cycles of 1.023 microseconds
>
>> The CPU has 64 clock periods of 14 * (1 / 14.318181 MHz) or 0.978µs and
>> one stretched period of 16 * (1 / 14.318181 MHz) or 1.117µs, which gives
>> an average clock period of 0.980µs. That works out to an average clock
>> speed of 1.0205 MHz.
>
> Ah, that explains the number in the comment. Doesn't explain why
> apparently nobody at Apple understood the relationship between MHz and microseconds. :-)
>

That comment has always annoyed me, too. I've seen the error quoted
numerous times. It must have mystified or misled countless people.

It's an example of why design reviews should include comments. ;-)

--
-michael - NadaNet 3.1 and AppleCrate II: http://michaeljmahon.com

---

Subject: How long will I WAIT?
Posted by Anonymous on Thu, 31 Oct 2019 16:07:43 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Wednesday, October 30, 2019 at 10:17:23 AM UTC-7, awanderin wrote:
> fadden <thefadden@gmail.com> writes:
>> If we consult Apple II Monitors Peeled page 85, it says the formula is:
>>
>>   2.5A**2 + 13.5A + 13 machine cycles of 1.023 microseconds

> The CPU has 64 clock periods of 14 * (1 / 14.318181 MHz) or 0.978µs and
> one stretched period of 16 * (1 / 14.318181 MHz) or 1.117µs, which gives
> an average clock period of 0.980µs. That works out to an average clock
> speed of 1.0205 MHz.
>
> I think the cycle count formula is 5 * a * a + 11 * a + 13, and that
> includes the JSR/RTS.

The code is:

```
fca8: 38        WAIT    sec             ;2
fca9: 48        WAIT2   pha             ;3
fcaa: e9 01     WAIT3   sbc     #$01    ;2
fcac: d0 fc             bne     WAIT3   ;2+
fcae: 68                pla             ;4
```

```
fcaf: e9 01             sbc    #$01      ;2
fcb1: d0 f6             bne    WAIT2     ;2+
fcb3: 60                rts              ;6
```

The inner loop is 5 cycles, except the last iteration which is 4. It doesn't execute A^2 times though, because A decrements each time. If initially A=4, it executes 4+3+2+1 times. So it's A*(A+1)/2 * 5 cycles.

The outer loop executes A times, and takes 12 cycles. We can compensate for counting 1 too many cycles on the last iteration of the inner loop (branch-not-taken takes one fewer) by subtracting one here. So that's A*11.

Outside of that, we have 8 cycles of non-loop stuff (SEC/RTS). Again, we're over-counting the last outer loop by 1 cycle, so we call it 7. If we want to add the JSR that called here that's another 6 cycles, but I prefer to put that in the caller's account instead.

So it's A*(A+1)/2 * 5 + A*11 + 7

Applying algebra:
(A*A/2 + A/2) * 5 + A*11 + 7
A*A*5/2 + A*5/2 + A*11 + 7
A*A*2.5 + A*13.5 + 7

Throw in the 6-cycle JSR and you get the formula from Apple II Monitors Peeled. So the cycle-count part of their formula is correct.

What's Where in the Apple gives the same formula (and also multiplies by "1..02 microseconds").

Somewhere along the way I picked up "(26 + 27*Acc + 5*(Acc*Acc))/2 cycles", which is mathematically equivalent. I like it a bit better because multiplying cycles by .5 just feels weird.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 19 Nov 2019 22:22:53 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.5.0-dev1 is now available. Key changes:

- Added non-unique local labels.
- Added label annotations.
- Added "go to last change" (moves selection to last edit).

Label annotations let you tack a '?' onto the end of a symbol. So if you're disassembling some code and see something that looks like it might hold high scores, but you're not sure, you can now give it the label "high_score?" instead of something like "high_score_maybe". The '?' is omitted when generating sources.

Non-unique local labels let you define labels with a scope that resets whenever a global variable is encountered. All supported cross-assemblers (64tass, ACME, cc65, Merlin 32) have something like this. For example:

```
first  ldx #$05
:loop  dex
       bne :loop

second ldy #$03
:loop  dey
       bne :loop
```

Because SourceGen is a disassembler and always knows which label goes with which address, you can write things that would gag an assembler:

```
:loop  ldx #$05
:loop  dex
       bne :loop
       dey
       bne :loop
       beq :done
global nop
:done  lda #$01
```

Labels will be adjusted as needed to generate code that assembles. In this case, the inner loop would become ":loop1", and ":done" would be promoted to a global label.

Newly-created labels now default to global. (They were previously "unique locals", meaning the code generator would try to make them be local, but insisted that they have a unique name.)

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 07 Dec 2019 21:22:21 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.5.0-dev2 is now available. Key changes:

- Added visualization generator interface to extension scripts.
- Added scripts for:
  - Apple II bitmaps, full-screen images, and bitmapped fonts
  - Atari 2600 sprites and playfields

- C64 sprites

Visualization generators take a bunch of parameters (file offset, width, height, etc) and generate graphical data. Thumbnail images are displayed in the code list. This makes it easier to figure out what a blob of data contains.

It looks like this: https://faddensoft.com/sgsample/vis.html

The peculiarities of the font embedded in the Budge 3D module (which is loaded on text page 1) become apparent once you have the whole thing laid out. The damage caused by the screen holes is visible in the glyphs in the rightmost column, and you can see that undamaged copies are repeated in the first 7 control characters.

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Thu, 26 Dec 2019 19:32:36 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.5.0-dev3 is now available. Key changes:

- Added animated visualizations.
- Added GIF generation to HTML exporter.
- Updated "set address" to work on a range of lines.
- Added options to "bulk" formatting.

Animated visualizations are constructed from other visualizations. For example, if you create a visualization for each frame of an animated figure running, you can create a list of the frames and play it. (This required changing the way visualizations are stored in the project file, so any created with the -dev2 release will disappear when the project is opened with -dev3.)

The HTML exporter outputs original-size GIF images. These are scaled up by the web browser. (Firefox doesn't seem to support nearest-neighbor scaling, so they may look a little fuzzy.) Animated visualizations are output as animated GIFs.

The "set address" feature lets you create an "ORG" directive that changes the address for the rest of the file. Sometimes you want to change the addresses for a range, e.g. code or data that gets relocated. If you select a range, "set address" will set the initial address as before, but will also create a second ORG directive at the bottom of the range that restores the address to the original file-load value.

The data operand editor's "bulk data" feature allows you to specify a maximum number of bytes per line. This is handy for data like bitmaps where you want one source line per row. You can also configure whether bulk data appears as a Merlin/ACME dense hex string, or a 64tass/cc65

series of comma-separated values. (The second item only affects the in-app display and HTML export; no effect on assembly source generation.)

I also added F6 as a shortcut to open the Project Symbols tab inside the Project Properties editor. And if you hover over the "recent project" buttons on the initial screen, the full project path appears in a tooltip.

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 27 Jan 2020 22:05:24 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.5 is now available. Key changes since v1.4:

- NOTE: extension script interfaces have changed. Pre-v1.5 plugins will not work with this release.
- Added visualization generation interface for converting embedded data to images.
  - Added extension scripts for C64 sprites, Atari 2600 graphics, and Apple II hi-res bitmaps, fonts, and shape tables.
  - Added animated bitmap visualizations.
  - Added GIF and animated GIF generation to HTML exporter.
- Added non-unique local labels (e.g. "@loop").
- Added uncertainty annotations to label (e.g. "score?").
- Added ability to set addresses on file chunks, making it easier to relocate sections of code.
- Added more options for "bulk" data formatting.
- Added file slicing and concatenation tools.
- Added "go to last change" feature.
- Updated project file formatting to make it more diff-friendly.

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 27 Jan 2020 22:08:28 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

The various commented disassemblies that were hosted at bigflake.com/disasm can now be found on https://6502disassembly.com/. As part of exercising SourceGen v1.5 I added a few more:

(1) Space Eggs, by Nasir Gebelli. A colorful shoot-em-up that effectively demonstrates the usefulness of visualizers. I made a short video: https://youtu.be/lSvEr5nCHbY

(2) Caverns of Freitag, by David Shapiro. An excellent example of mixing Applesoft with assembly language. I used a custom visualizer to format the maze map. Adding visualizers for the tile set elements revealed graphics for monsters that didn't make it into the game.

(3) Starship Commander, by Gilman Louie. The game is almost entirely Applesoft, so doesn't have a whole lot for SourceGen to do. I disassembled the BASIC portions a few years back, and figured I'd share, so I took the HRCG apart and converted the fonts.

(4) Adventure, for the Atari 2600, by Warren Robinett. This is a "port" of a disassembly floating around the Internet, mostly to confirm that address mirroring and 2600 graphics work.

FWIW, SourceGen v1.5 has what I consider to be the full basic feature set for a disassembler. It still needs some features and has some rough edges, but it flows pretty well now.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sat, 08 Feb 2020 23:05:22 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

I was feeling nostalgic this afternoon.

  https://6502disassembly.com/a2-applevision/

Today I learned that AppleVision randomizes the dance moves if you let it repeat.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Hugh Hood on Sun, 09 Feb 2020 01:20:52 GMT
View Forum Message <> Reply to Message

You did all that in an afternoon?

You're good.

On 2/8/2020 5:05 PM, fadden wrote:
> I was feeling nostalgic this afternoon.
>
> https://6502disassembly.com/a2-applevision/
>

> Today I learned that AppleVision randomizes the dance moves if you let it repeat.
>

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Sun, 09 Feb 2020 05:55:31 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Saturday, February 8, 2020 at 5:20:58 PM UTC-8, Hugh Hood wrote:
> You did all that in an afternoon?

Somewhere in my many boxes of my Apple II stuff is my original disassembly of SSI's RDOS. Printed out on tractor feed paper, colored with highlighter pens, scribbles all over. Took ages.

A few years back I tried digging into the Budge 3D stuff and parts of a game by generating a disassembly with CiderPress and digging through it in a text editor. That works okay for figuring out generally where things are, but unless you're doing search & replace on functions and variables things don't fall into place as easily.

For AppleVision I extracted files and generated BASIC listings with CiderPress, then dug through the binary with SourceGen using the CALL addresses as entry points. I had most of it done in about two hours. (Note to self: don't start digging into something right before lunch.) Spent another hour or so polishing it up, writing the descriptive HTML page, generating a screen shot, etc.

The code is pretty straightforward, but this would've taken a lot longer without the tools.

Of course, if you consider how much time I've spent on CiderPress and SourceGen, I'm probably not coming out ahead...

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Michael AppleWin Debu on Mon, 17 Feb 2020 15:52:07 GMT
View Forum Message <> Reply to Message

On Monday, January 27, 2020 at 2:05:26 PM UTC-8, fadden wrote:
> 6502bench SourceGen v1.5 is now available. Key changes since v1.4:

Hey Andy

I've been meaning to reply for a while -- thanks for this disassembler! I'm starting to use this more and more for a few back-logged projects and SourceGen is quickly becoming one of my favorite tools to use! It doesn't suck out-of-the-box so I give it a "One-thumbs-up!" :-)

I noticed there are a few minor rough spots that prevents it from getting my full Two-thumbs-up

though:

* The learning curve is a little stepper then I would have liked. I believe it just needs a "Quick Start" section / guide that covers a detailed step-by-step game disassembly showing code + font + sprites. I see there are a few tutorials, and while they do provide some step-by-step instructions,
  a) the step-by-step instructions aren't labeled as such, and
  b) don't include step-by-instructions **with** pictures showing where to click and what the UI should look like.

* HGR Visualizer really needs NTSC support with 12-bit graphics. :-) Another drop-down would be good. Looks like RenderBitmap() should be extended with a few more ColorMode enums. SetHiResPalette() probably needs to be extended to support 4-phase colors.

* I noticed that DHGR is missing. Any plans to add support for this?

* The help table-of-content headers doesn't mention keywords such as:
  * HGR
  * Font
  * Ripping, or
  * Sprite
  * Game

This makes it much harder to start as it isn't intuitive that they would be under "Visualizations"

I would be more then happy to make SourceGen more user friendly and provide feature requests for the above. Should I submit PRs (Pull Requests) on GitHub?

Thanks again for a neat utility!

Michael

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Mon, 17 Feb 2020 23:38:12 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Monday, February 17, 2020 at 7:52:09 AM UTC-8, Michael 'AppleWin Debugger Dev' wrote:
> * The learning curve is a little stepper then I would have liked. I believe it just needs a "Quick Start" section / guide that covers a detailed step-by-step game disassembly showing code + font + sprites. I see there are a few tutorials, and while they do provide some step-by-step instructions,
> a) the step-by-step instructions aren't labeled as such, and
> b) don't include step-by-instructions **with** pictures showing where to click and what the UI should look like.

I agree. The current documentation is really a reference manual. During the last chunk of development I realized that it needed a separate "getting started" sort of section, but didn't get

around to writing one.

Do people still read? Maybe a tutorial video instead? The original pre-1.0 demo is more or less the first tutorial (https://youtu.be/dalISyBPQq8), although things have changed quite a bit since Sep 2018. I tried doing a longer video showing the disassembly process, but after 20 minutes I determined that I was extremely dull and gave up. (https://youtu.be/WZvrMOzyHhs)

> * HGR Visualizer really needs NTSC support with 12-bit graphics. :-) Another drop-down would be good. Looks like RenderBitmap() should be extended with a few more ColorMode enums. SetHiResPalette() probably needs to be extended to support 4-phase colors.
>
> * I noticed that DHGR is missing. Any plans to add support for this?

I thought about getting fancier with the converter, but at the end of the day SourceGen is a disassembler, not a graphics converter. The primary goal is to let the user see what the graphics are, not generate camera-ready copy, so I don't do half-pixel shifts or odd color fringes for hi-res data. (Most people will only see the output as a 64x64-ish GIF embedded in an HTML listing anyway.) I figured anybody who wanted something beyond basic would roll their own.

Bear in mind that nothing specific to the Apple II is compiled into SourceGen; everything is loaded at run time. So if you want to do fancy visualizers you can tweak the code without needing any development tools. (It does make life a *lot* easier if you do everything in Visual Studio though... syntax checking et.al.) It also means that you can substitute visualizers globally (by updating your RuntimeData copy) or per-project. I created a couple of visualizers specifically for Caverns of Freitag data (very easy way to render the full 80x80 map).

Re: DHGR, there aren't that many programs I can think of that have embedded DHGR graphics (Airheart is one example, drawing programs like Dazzle Draw would be another), so it hasn't been a priority.

> I would be more then happy to make SourceGen more user friendly and provide feature requests for the above. Should I submit PRs (Pull Requests) on GitHub?

Contributions are always welcome. If you want to submit High Quality graphics converters they should be in a separate .cs, either with the same gen string (to allow drop-in replacement) or different (to allow both to co-exist in a project for an A/B comparison)... haven't thought through which makes the most sense.

> Thanks again for a neat utility!

Thanks for the AppleWin debugger. :-)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Steven Hirsch on Tue, 18 Feb 2020 13:17:28 GMT
View Forum Message <> Reply to Message

On 2/17/20 6:38 PM, fadden wrote:

> Do people still read? Maybe a tutorial video instead? The original
> pre-1.0 demo is more or less the first tutorial
> (https://youtu.be/dalISyBPQq8), although things have changed quite a bit
> since Sep 2018. I tried doing a longer video showing the disassembly
> process, but after 20 minutes I determined that I was extremely dull and
> gave up. (https://youtu.be/WZvrMOzyHhs)

Exactly this. I cannot stand video instructions. Nothing better than a well thought-out manual in my book.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Michael AppleWin Debu on Tue, 18 Feb 2020 14:41:42 GMT
View Forum Message <> Reply to Message

On Tuesday, February 18, 2020 at 5:17:34 AM UTC-8, Steven Hirsch wrote:
> On 2/17/20 6:38 PM, fadden wrote:
>
>> Do people still read? Maybe a tutorial video instead? The original
>> pre-1.0 demo is more or less the first tutorial
>> (https://youtu.be/dalISyBPQq8), although things have changed quite a bit
>> since Sep 2018. I tried doing a longer video showing the disassembly
>> process, but after 20 minutes I determined that I was extremely dull and
>> gave up. (https://youtu.be/WZvrMOzyHhs)
>
> Exactly this. I cannot stand video instructions. Nothing better than a well
> thought-out manual in my book.

Agreed. Text manuals are my first choice, videos are my second choice (given the lack of a text manual.)

Both are good at presenting information in a linear fashion but for random access / searching videos SUCK when you are looking for a specific topic. Ctrl-F is much, much faster and easier.

I've forked the repo and will be adding a detailed step-by-step tutorial to it using an actual game that encompasses almost every thing a "real" disassembly would need -- since it _is_ based on an actual disassembly. :-)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 18 Feb 2020 16:44:28 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Tuesday, February 18, 2020 at 6:41:44 AM UTC-8, Michael 'AppleWin Debugger Dev' wrote:

> I've forked the repo and will be adding a detailed step-by-step tutorial to it using an actual game that encompasses almost every thing a "real" disassembly would need -- since it _is_ based on an actual disassembly. :-)

I've been pretty careful to this point to avoid checking in somebody else's copyrighted code to the 6502bench repository. The existing tutorials and examples are all code I've written, for reasons other than mere vanity. :-)

If you do use a commercial game as the subject, and it hasn't been explicitly released in some way by the copyright holder, then including the game itself with the tutorial files could be awkward. Hosting the binary somewhere else (say, 6502disassembly.com) would be fine, and I figure any screen shots or other references to parts of the file are "fair use". But I'm trying to keep the 6502bench repository relatively clean.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Michael AppleWin Debu on Tue, 18 Feb 2020 17:35:23 GMT
View Forum Message <> Reply to Message

On Tuesday, February 18, 2020 at 8:44:31 AM UTC-8, fadden wrote:
> I've been pretty careful to this point to avoid checking in somebody else's copyrighted code to the 6502bench repository. The existing tutorials and examples are all code I've written, for reasons other than mere vanity. :-)
>
> If you do use a commercial game as the subject, and it hasn't been explicitly released in some way by the copyright holder, then including the game itself with the tutorial files could be awkward. Hosting the binary somewhere else (say, 6502disassembly.com) would be fine, and I figure any screen shots or other references to parts of the file are "fair use". But I'm trying to keep the 6502bench repository relatively clean.

That's perfectly understandably to avoid the retarded and utterly broken Copyright system. The first step will be something along the lines of:

1. Download Lode Runner. (Link)

From Wikipedia looks like "Tozai Games currently holds the copyright and trademark rights". Shame that Doug E. Smith's widow didn't release the source code for Lode Runner upon her husband's death.

This is fucking retarded that copyright holds [past] culture hostage -- but that is a discussion for another day. At least we can work around the problem by making someone else responsible. =P

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Michael AppleWin Debu on Tue, 18 Feb 2020 17:45:53 GMT

View Forum Message <> Reply to Message

On Monday, February 17, 2020 at 3:38:14 PM UTC-8, fadden wrote:

> Do people still read?

Yes! Not everyone is a millennial. =P

> Contributions are always welcome.

I created a master/uber wish-list with issue #53. Many of these are easy low-hanging fruit. I'll get C# installed so I can start fixing the trivial ones in my fork and issue PRs (Pull Requests).

> If you want to submit High Quality graphics converters they should be in a separate .cs, either with the same gen string (to allow drop-in replacement) or different (to allow both to co-exist in a project for an A/B comparison)... haven't thought through which makes the most sense.

Understood. NTSC quality isn't "required" right now -- more of a long-term QoL.

>> Thanks again for a neat utility!
> Thanks for the AppleWin debugger. :-)

Ideally all of 6502bench's functionality should ben built into the debugger.. This gives me a great list of features to consider embedding natively. :-)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Wed, 19 Feb 2020 00:58:00 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

On Tuesday, February 18, 2020 at 9:45:54 AM UTC-8, Michael 'AppleWin Debugger Dev' wrote:
> I created a master/uber wish-list with issue #53. Many of these are easy low-hanging fruit. I'll get C# installed so I can start fixing the trivial ones in my fork and issue PRs (Pull Requests).

I saw bugs filed and fixed the easy ones. :-)

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Michael AppleWin Debu on Wed, 19 Feb 2020 17:06:11 GMT
View Forum Message <> Reply to Message

On Tuesday, February 18, 2020 at 4:58:02 PM UTC-8, fadden wrote:
> On Tuesday, February 18, 2020 at 9:45:54 AM UTC-8, Michael 'AppleWin Debugger Dev' wrote:

>> I created a master/uber wish-list with issue #53. Many of these are easy low-hanging fruit. I'll get C# installed so I can start fixing the trivial ones in my fork and issue PRs (Pull Requests).
>
> I saw bugs filed and fixed the easy ones. :-)

Thanks for the fixes! Much appreciated!

Now that I'm able to build it locally (#78) I'm in the process of verifying each of them -- that way I don't have to wait for an official release.

I also added category section headers to the master list. I'll take care of the documentation stuff I listed and create PRs once they are ready.

Lode Runner has been a good exercise in "functionality coverage" !

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Thu, 12 Mar 2020 17:37:56 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.6-dev1 is now available. Key changes since v1.5:

- Added visualization of wireframe meshes.
- Added "sprite sheets" to hi-res visualizer.

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

I updated the Budge 3D disassembly to use the wireframe feature. If you're curious to see what that looks like, go to https://6502disassembly.com/a2-budge3d/MODULE.SHIP.CUBE.html #SymNumObjects and scroll up a couple of lines.

I have some more interesting projects but those are still works in progress.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 31 Mar 2020 00:10:03 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.6 is now available. Key changes since v1.5:

- Added support for wireframe visualizations.

- Added custom colors for Notes.
- Added "sprite sheets" to the Apple II bitmap visualizer.

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases

---

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Tue, 31 Mar 2020 00:28:32 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

I added two new projects to the 6502 disassembly site:

Elite: https://6502disassembly.com/a2-elite/
Stellar 7: https://6502disassembly.com/a2-stellar7/

Elite is enormous and very similar to well-documented versions on other platforms, so large sections of code are uncommented. I focused on Apple II-specific stuff and figuring out how to render the shapes. It turns out some of the shape data is wrong and causes visible glitches when you render at higher resolutions, but I was able to work around it in the visualizer.

Stellar 7 is complete (unless I overlooked something). The line drawing code (https://6502disassembly.com/a2-stellar7/ROCK1.html#SymDrawLi ne) is remarkable -- run-slice algorithm with 3 different functions for horizontal lines to take advantage of updating multiple pixels per loop. I suspect the calculation of the slope wrecks performance for short lines though.

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Antoine Vignau on Tue, 31 Mar 2020 11:20:02 GMT
View Forum Message <> Reply to Message

That is cool, Andy!

Subject: Re: 6502bench SourceGen disassembler updated
Posted by Anonymous on Fri, 15 May 2020 20:15:22 GMT
View Forum Message <> Reply to Message

Originally posted by: fadden

6502bench SourceGen v1.7-dev1 is now available. Key changes since v1.6:

- Added visualization generator for Atari AVG.
- Added visualization generator for NES pattern tables.

Most of you are probably familiar with the NES (Nintendo Entertainment System), at least in passing. As an exercise, I ported a disassembly of Super Mario Bros.

https://6502disassembly.com/nes-smb/

I suspect relatively few of you are familiar with Atari's AVG (Analog Vector Graphics). It was used in a bunch of Atari video arcade games back in the early 1980s. The one that caught my eye ~40 years ago was called Battlezone. When I learned a few years back that it was written for the 6502, I decided to disassemble it.

https://6502disassembly.com/va-battlezone/

Neither of these is Apple II-related (apologies!). I'm somewhat tempted to pry open the Atarisoft version of Battlezone to see if they shared any code.

The project web site is https://6502bench.com/. Source code and pre-built Windows binaries are available from https://github.com/fadden/6502bench/releases