Digital AV –Record layouts & File inventory

2018 Edition – SDK Revision: I-728

AV-Writ.DX2 (uint16 + 2 × bytes = 32 bits)

Record 0 bits	Caps 2 bits	WordKey 14 bits		Punctuation 8 bits	Transitions 8 bits
0	0x8___	0x0015	(in)	0x00	0xE0
1	0x0___	0x0136	(the)	0x00	0x00
2	0x0___	0x24F9	(beginning)	0x00	0x00
…		<< Beginning of Genesis 1 depicted above >>			
BDDB9	0x8___	0x0136	(the)	0x00	0xE0
BDDBA	0x8___	0x2CB2	(revelation)	0x00	0x00
BDDBB	0x0___	0x001D	(of)	0x00	0x00
…		<< Beginning of Revelation 1 depicted above >>			
C0C91	0x0___	0x015C	(you)	0x00	0x00
C0C92	0x0___	0x0036	(all)	0xE0	0x00
C0C93	0x8___	0x018A	(amen)	0xE0	0xF0
		<< End of Revelation 22:21 depicted above >>			

Punctuation Byte

Description	Bits
PUNC::clause	0xE0
PUNC::exclamatory	0x80
PUNC::interrogative	0xC0
PUNC::declarative	0xE0
PUNC::dash	0xA0
PUNC::semicolon	0x20
PUNC::comma	0x40
PUNC::colon	0x60
PUNC::possessive	0x10
PUNC::closeParen	0x0C
MODE::parenthetical	0x04
MODE::italics	0x02
MODE::Jesus	0x01

Capitalization bits and WordKey (UInt16)

Description	Bit Pattern (Hex)
English Word	0x3FFF (This mask produces lookup key for word)
1st Letter Cap	0x8000 (example: Lord)
All Letters	0x4000 (example: LORD)

File extents are representative of the format and the corresponding record width for files containing fixed width records. Extents of binary-formatted files begin with one of: {DX = for data; IX = for indices; VLT = for variable-length tables}; for three-letter extents, the last digit contains the count of 16-bit segments as a single hex digit. The size digit in the file extent is significant merely as a reminder to the developer, but the AV-Writ file actually has three variants; and each variant is identified by its file extent: .DX2, .DX3, and .DX8. Each data file facilitates rendering bible text. However, the developer is expected to choose just one variant. Two additional extents are VLT for "variable-length table" and DB for "Sqlite database file".

AV-Writ.DX3 (uint16 + 2 x bytes + uint16 = 48 bits)

Record 0 bits	Caps 2 bits	WordKey 14 bits		Punctuation 8 bits	Transition 8 bits	Part-of-Speech 16 bits
0	0x8___	0x0015	(in)	0x00	0xE0	0x0400
1	0x0___	0x0136	(the)	0x00	0x00	0x0D00
2	0x0___	0x24F9	(beginning)	0x00	0x00	0x4010
…	<< Beginning of Genesis 1 depicted above >>					
BDDB9	0x8___	0x0136	(the)	0x00	0xE0	0x0D00
BDDBA	0x8___	0x2CB2	(revelation)	0x00	0x00	0x5010
BDDBB	0x0___	0x001D	(of)	0x00	0x00	0x0400
…	<< Beginning of Revelation 1 depicted above >>					
C0C91	0x0___	0x015C	(you)	0x00	0x00	0xA2B0
C0C92	0x0___	0x0036	(all)	0xE0	0x00	0x0D00
C0C93	0x8___	0x018A	(amen)	0xE0	0xF0	0x8010
	<< End of Revelation 22:21 depicted above >>					

As first mentioned on the previous page, the final hex-digit that follows the DX_ in the file extent identifies the width of records for the content. That digit is the count of 16-bit segments per record. For example, DX8 contains 8 16-bit segments per record. This multiplies out to 128 bits, which equates to a fixed record width of 16 bytes. To be clear, DX8 does not mean every field is 16-bits; it is just a convenient shorthand for depicting the record width for files containing fixed-width records.

Person/Number (4 bits)

Description	Nibble Patterns	
Person bits	0x3	(0b0011)
Number bits	0xC	(0b1100)
1st Person	0x1	(0b0001)
2nd Person	0x2	(0b0010)
3rd Person	0x3	(0b0011)
Singular	0x4	(0b0100)
Plural	0x8	(0b1000)
WH*	0xC	(0b1100)

The Person/Number (PN) is left-most nibble of the Part-of-Speech field in the larger AV-Writ record formats (DX3 & DX8). PN applies to pronouns and associated archaic verb forms. For instance, **thy** is second-person singular whereas Early Modern English **you** is always plural. Personal pronouns in the AV-SDK are marked for both person and number.

Transition (4 bits)

Description	Low Nibble
EndBit	0x1_
BeginingOfVerse	0x2_
EndOfVerse	0x3_
BeginingOfChapter	0x6_
EndOfChapter	0x7_
BeginingOfBook	0xE_
EndOfBook	0xF_

Transition bits are mostly just a convenience, as this information is redundant with information contained within index files (IX4 & IX2). However, the right-nibble is not redundant: it contains a zero-based index of the sentence of the verse which contains the token. Many verses contain only a single sentence, but some contain more. Sentence index is identified by the NLTK toolkit. Sentence index is always zero through fifteen (a nibble of data).

Digital AV –Record layouts & File inventory
2018 Edition – SDK Revision: I-728

AV-Writ.DX8 (4 × uint16 + 2 × uint16 + 2 × byte + uint16; 128 bits)

Record 0 bits	Hebrew/Greek (4 x 16 bits)	Verse Index 16 bits	Caps 2 bits	WordKey 14 bits	Punctuation 8 bits	Transition 8 bits	Part-of-Speech 16 bits
0	0x391C 0x0 0x0 0x0	0x0000	0x8___	0x0015 (in)	0x00	0xE0	0x0400
1	0x391C 0x0 0x0 0x0	0x0000	0x0___	0x0136 (the)	0x00	0x00	0x0D00
2	0x391C 0x0 0x0 0x0	0x0000	0x0___	0x24F9 (beginning)	0x00	0x00	0x4010
…	<< Beginning of Genesis 1 depicted above >>						
BDDB9	0x25A0 0x0 0x0 0x0		0x8___	0x0136 (the)	0x00	0xE0	0x0D00
BDDBA	0x25A0 0x0 0x0 0x0		0x8___	0x2CB2 (revelation)	0x00	0x00	0x5010
BDDBB	0x0978 0x0 0x0 0x0		0x0___	0x001D (of)	0x00	0x00	0x0400
…	<< Beginning of Revelation 1 depicted above >>						
C0C91	0x1460 0x0 0x0 0x0	0x797B	0x0___	0x015C (you)	0x00	0x00	0xA2B0
C0C92	0x0F74 0x0 0x0 0x0	0x797C	0x0___	0x0036 (all)	0xE0	0x00	0x0D00
C0C93	0x0119 0x0 0x0 0x0	0x797D	0x8___	0x018A (amen)	0xE0	0xF0	0x8010
	<< End of Revelation 22:21 depicted above >>						

Hebrew/Greek (4 x bytes; 32 bits)

Strongs #1	Strongs #2	Strongs #3	Strongs #4
1st Strongs #	2nd Strongs #	3rd Strongs #	4th Strongs #

POS (16 bits)

Noun	0x_01_
Pronoun	0x_03_
Pronoun: Neuter	0x_031
Pronoun: Masculine	0x_032
Pronoun: Non-feminine*	0x_033
Pronoun: Feminine	0x_034
Pronoun: Genitive	0x_038
Pronoun: Nominative	0x_07_
Pronoun: Oblique	0x_0B_
Pronoun: Reflexive	0x_0F_
Verb	0x_10_
to	0x0200
Preposition	0x0400
Interjection	0x0800
Adjective	0x_A00
Adverb	0x_B00
Determiner	0x0D_0
Participle	0x1D00

The DX8 format augments the abbreviated AV-Writ record formats by adding a pointer to the AV-Verse index file. It also adds 1 to 4 Strong's numbers per word position. Strong's numbers are an integer representation of the original Hebrew/Greek words from which the English words were originally translated (Refer to the Strong's Exhaustive Concordance for additional backround information).

It should be noted that while words in the Old Testament can have a maximum of four Strong's numbers representing the Hebrew associated with a single English word. The New Testament can only have a maximum of three Strong's numbers representing the Greek associated with a single English word. This is characteristic of the KJV translation and does not of itself suggest inherent differences between Greek and Hebrew. POS bits, docemented on the right in summary form, are more fully documented in a separate stand-alone Part-of-Speech SDK document.

> * *his* is used ambiguously in the Authorized Version for third-person-singular pronouns. *his* can mean either masculine or neuter (*its* appears just once). Therefore, we cannot mark *his* uniformly as masculine or neuter. Instead, we mark *his* as non-feminine.

Digital AV –Record layouts & File inventory
2018 Edition – SDK Revision: I-728

AV-Writ.ascii (null-terminated/space-separated records: 56+ chars/line)

Record 5 digits	Hebrew/Greek 16 digits	V-Index 4 zigits	Caps 1 digit	WordKey 4 digits	Punctuation 2 digits	Trans 2 digits	POS 4 digits	BCVW 8 digits	Text: LF-terminated
00000	391C000000000000	0000	8	0015	00	E0	0400	01010101	In\0a
00001	391C000000000000	0000	0	0136	00	00	0D00	01010102	the\0a
00002	391C000000000000	0000	0	24F9	00	00	4010	01010103	beginning\0a
…	<< Beginning of Genesis 1 depicted above >>								
BDDB9	025A000000000000	?	8	0136	00	E0	0D00	42010101	The\0a
BDDBA	025A000000000000	?	8	2CB2	00	00	5010	42010102	Revelation\0a
BDDBB	0978000000000000	?	0	001D	00	00	0400	42010103	of\0a
…	<< Beginning of Revelation 1 depicted above >>								
C0C91	1460000000000000	797B	0	015C	00	00	A2B0	4216150A	you\0a
C0C92	0F74000000000000	797C	0	0036	E0	00	0D00	4216150B	all\0a
C0C93	0119000000000000	797D	8	018A	E0	F0	8010	4216150C	Amen\0a
	<< End of Revelation 22:21 depicted above >>								

The ascii variations of SDK files are intended to be informative, and are not considered to be core components of the inventory. Non-optimal performance would be achieved if a choice were made to develop against the ascii files instead of the binary formats. Yet, they are provided in the SDK to illuminate the formats of similarly organized binary files. In ASCII represntation, the punctuation byte does not yet set the 0x8 bit for PUNC::closeParen (This bit is new for the HC11 release).

AV-Book.IX4 (UInt16 + 2 × byte + 32 ASCII characters = 64 bits)

Record 0 bits	Chapter Index 16 bits	Book Number 8 bits	Chapter Count 8 bits	Book Names 32 ascii characters (slash-separated & null-padded)
0	0x000	1	50	Genesis/Gen/Ge\0\0\0 … \0
1		2	40	Exodus/Exo/Ex\0\0\0 … \0
2		3	27	Leviticus/Lev/Le\0\0\0 … \0
…				
65	0x4A4	66	22	Revelation/Rev/Re/Rv\0\0\0 … \0

Digital AV –Record layouts & File inventory

2018 Edition – SDK Revision: I-728

AV-Chapter.IX4 (UInt32 + 2 × UInt16 = 64 bits)

Record 0 bits	Bible Index 32 bits	Verse Index 16 bits	Word Count 16 bits
0x000 (genesis:1)	0x00000	0x0000	0x31D
0x001 (genesis:2)	0x0031D	0x001F	0x278
0x002 (genesis:3)	0x00595	0x0038	0x2B7
	. . .		
0x4A2 (revelation:20)	0xC0769	0x797B	0x1DD
0x4A3 (revelation:21)	0xC0A56	0x797C	0x2ED
0x4A4 (revelation:21)	0xC0C93	0x797D	0x23D

AV-Verse.IX2 (4 × byte = 32 bits)

Record# 0 bytes	Book, Chapter, Verse, Words 32 bits: BB:CC:VV:WordCnt	
0x0000	0x0101010A	In the beginning …
0x0001	0x0101021D	And the Earth …
0x0002	0x0101030A	And God said …
	...	
0x797B	0x4216152C	And if any man … are written in this book.
0x797C	0x42161510	He which testifieth … Even so, come, Lord Jesus.
0x797D	0x4216150C	The grace of our Lord … be with you all. Amen

Digital AV –Record layouts & File inventory
2018 Edition – SDK Revision: I-728

AV-Lexicon.VLT (variable length table; replaceable by AVX-Lexicon.VLT or AVX-Lexicon.DB)

WordSize 2 bytes (16 bits)	WordCnt 2 bytes (16 bits)	Null-separated character arrays WordCnt*2 bytes
1	3	'a\0i\0o\0' [keys = 1,2,3]
2	40	'ah\0ai\0am\0 ... ye\0' [keys = 4,5,6, ... ,353]
3	311	'abi\0act\0add\0 ... zur\0' [keys = 354,355,356, ... ,1440]
…		
18	2	'jonathelemrechokim\0maher-shalal-hash-baz\0' [keys = 12565, 12566]
0	12567	(total)

AV-Lexicon.VLT is the most compact of the three variant lexicons. While these three lexicons are interchangeable, but AV-Lexicon.VLT contains only the display word, whereas, the two alternate lexicons contain additional lexical information beyond just the key and display value. AVX-Lexicon.DB introduces a dependency on an additional sqlite library. For this reason, AV-Lexicon.VLT and AVX-Lexicon.VLT are the preferred lexicons. A usage example for AV-Lexicon,VLT is provided in avtext.go. A usage example for AVX-Lexicon is provided in avx.go (available on github).

AVX-Lexicon.DB (sqlite database)

AV-Lexicon.db is an Sqlite 3.x database; here is the schema for the lexicon:

```
CREATE TABLE Lexicon
(
        key         INTEGER NOT NULL,
        search      string NOT NULL,
        display     string,
        modern      string,
        pos         string,
        person      INTEGER NOT NULL,
        number      INTEGER NOT NULL,
        isname      INTEGER NOT NULL
);
CREATE UNIQUE INDEX LexKey_idx ON Lexicon(key);
CREATE UNIQUE INDEX LexSearch_idx ON Lexicon(search);
```

Digital AV –Record layouts & File inventory

2018 Edition – SDK Revision: I-728

AVX-Lexicon.VLT (variable length table)

Size 1 byte	Search Size bytes	Size 1 byte	Display Size bytes	Size 1 byte	Modern Size bytes	POS 2 bytes	Name/Person/Num 1 byte
1	a	0		0			0x00
1	i	1	I	0			
1	o	0		2	oh		
…							
4	thou	0		3	you		
…							
18	mahershalalhashbaz	21	maher-shalal-hash-baz	0			
0							

Implicit key on the above table is merely the order of the records. The key is one-based. Accordingly, the record key for "a" is 1. It is noteworthy that keys on AV-Lexicon.VLT, AVX-Lexicon.VLT, and AVX-Lexicon.DB are 100% compatible.

AVX-Lemma.VLT (variable length table)

Size 1 byte	Lemma Size bytes	POS 2 bits	wordkey 14 bits
2	be		
4	went		
3	run		
…			
0			

The AVX-Lemma file is new with the HC11 revision. Its data is best ingested with a HashMap: The Lemma would not be the unique key to the HashMap, as many wordkeys resolve to the same Lemma (e.g. The lemma 'be' corresponds to 'are', 'were', 'is', and 'be'). Moreover, as words like 'run' can function both as a verb and a noun, the HashMap-key to the lemma should incorporate the POS bits or'ed with the wordkey. As English exhibits limited inflections, lemmas are captured only for four basic parts-of-speech (identified below with the corresponding bit-pattern):

0x0000:	Noun
0x4000:	Verb
0xC000:	Adverb
0x8000:	Adjective

Digital AV –Record layouts & File inventory

2018 Edition – SDK Revision: I-728

avtext.go (golang source code)

AVtext.go implements a web-server (HTTP server) that provides the entire text of the AV bible using simple semantics. As the web-server is not hardened, it should be placed behind a reverse-proxy if exposed to the open Internet. This is a common pattern and Caddy, a more general-purpose web-server, can be easily configured as a reverse-proxy. On to the details about avtext.go …

There are a couple of URLs used for testing and validation. They also illustrate how avtext.go can be extended:

- http://localhost:2121/
- http://localhost:2121/help
- http://localhost:2121/validate

> *Example of GoLang source in operation at avbible.net:*
>
> https://avbible.net/av/
> (the web-site above also utilizes Caddy as a reverse-proxy for HTTPS)

The / endpoint simply reports the release number of the optional avtext.go web-server component. The /help endpoint provides primitive information about the web-service. /help can be easily replaced by developer. The /validate endpoint reports on the validity of data files in accordance with the bom (The "bom", or bill of materials, is described in the section labelled AV-Inventory.bom later in this document. In addition to the administrative URL's described above, here is a list of the foundational endpoints that provide the core functionality of avtext.go:

1. http://localhost:2121/genesis
2. http://localhost:2121/genesis/1
3. http://localhost:2121/gen/1?sessionID
4. http://localhost:2121/rev/22?sessionID=day&amen
5. http://localhost:2121/rev/22?sessionID=$FFFFFFFFFFFF
6. http://localhost:2121/css/sessionID.css

All of these endpoints can be summarized as one of two types: getting the chapter of a book, or getting a CSS stylesheet. When no chapter is provided, chapter 1 is always implied. When no session identifier is provided, the resulting chapter request is decorated with the baseline stylesheet, named /css/AV-Stylesheet.css. When a session identifier is provided, the session number dictates the name of the CSS file that will decorate the chapter request. Moreover, avtext.go can compile information into a CSS stylesheet. When a request is made for Genesis using the URL depicted in #3 above, a stylesheet becomes linked in the response to a stylesheet with the URL depicted in #8 above. A web-browser will make an immediate subsequent request to get the stylesheet. If /css/sessionID.css does not exists, avtext.go will automatically compile a file named /css/sessionID.avspec. Similarly, but easier to understand in #4 above, the URL would generate CSS which would highlight the words **day** and **amen**. In order to maintain optimal performance, session identifiers are non-volatile. In order to overwrite a *.css files and/or *.avspec files, they must be manually deleted beforehand.

URL form #3 and #5 are discussed under the description of the *.avspec format.

Digital AV –Record layouts & File inventory

2018 Edition – SDK Revision: I-728

*.avspec file format

WordKey Count UInt16	Array of UInt16	
0xnnnn	0xnnnn Count of WordKeys is followed by WordKey list [corresponds to AV-Lexicon]	

BookChapter UInt16	Verse Count byte *(matching verses)*	Array of byte
0xbbcc	0xkk	0xkk Count of matching verses is followed by an array of Verse numbers
…		
0xbbcc	0xjj	0xjj Count of matching verses is followed by an array of Verse numbers
0x0000		

AVtext.go software ignores everything after the first record above. Only that first record defines the CSS file. And that first line is expanded word-for-word into highlights for each supplied wordkey. A slight variation here is that Strong's numbers will eventually also support highlighting. To highlight Strong's numbers, set the 0x8000 bit for Hebrew and the 0x4000 bit for Greek.   The URL form that was depicted with this syntax, sessionID=$FFFFFFFFFFFF, is primarily intended for testing. Here, the hex digits that follow the dollar sign ($) are expected to be expansions of the format described above (No record separators, just a representation of the raw bytes described above, in Big-Endian order).

AV-Stylesheet.css (text file containing CSS for avtext.go; optional)

This standard-format CSS stylesheet should be included when avtext.go is utilized in your development. This optional stylesheet is included in the SDK, but it can be customized in any way by the web designer. However, the web designer should realize that any references in the CSS to image files will result in 404 errors unless support is explicitly added to avtext.go by your development team. Finally, avtext.go always links chapter output to the AV-Stylesheet.css stylesheet, even when a *.avspec derived stylesheet is also specified.

Digital AV –Record layouts & File inventory

2018 Edition – SDK Revision: I-728

```
REVISION IDENTIFIERS
Digital-AV SDK:        I-728
AVX-extensions:        H-C14
SDK Document:          I-814
```

AV-Inventory.bom (text file which identifies core inventory; optional)

This is a text file that identifies the release and the delivered files for any given release. "bom" stands for bill of materials. For each artifact of the SDK, the bom lists each filename along with its corresponding MD5. The avtext.go server implements a validation function that will read this file and report if the MD5's agree with the SDK files on disk. This way corruption can is averted and as a revision can be validated in an automated fashion. Each Plate revision is released with a bom that should be deployed with the SDK. By doing this, streamlined plate revision checks are straightforward. MD5's are calculated both on core SDK files, and on optional components, but not upon the bom itself. Please note that the bom is not updated on every Alpha/Beta release, but can be updated manually using avtext.go. After downloading AV-Inventory.bom, it is recommended that you copy each bom to a name that will not be overwritten. For each revision, it is recommended that you would copy AV-Inventory.bom to another location (e.g. AV-Inventory.HC11) along with the documentation associated with that release [this document may change over time].

OVERALL PROJECT STATUS:

The #HA29 revision represented the first official Release Candidate (RC1) of the 2017 SDK. The #HA29 revision and subsequent revisions differ from the previous #HA10 revision in that transition bits now include sentence-index and POS. Transition bits and POS bits prior to the #HA29 revision, are not reliable. As of the #HA10 revision, there is no longer a dependency upon the 2011 edition of the SDK. All subsequent revisions are expected to be based entirely upon an earlier baseline revision of the 2017 SDK. Release 2017.HC11 represents the latest baseline release.

It's an exciting time at AV Text Ministries, and if you want to lend a hand, let us know your technical skills and interests and we can help jumpstart you onto the team. Currently, AV Text Ministries is 100% volunteer, so if you don't just have passion about the mission as your raw motivation, it might not be the best fit.

Finally, on the non-technical side of things, we would certainly welcome a ministry sponsor that would want to place AV Text Ministries under the banner of their own local church ministry. Check out http://ReceivedText.com, http://Digital-AV.org, and http://avtext.org to discover our overall vision.

HOW THE DIGITAL-AV "PLATES" WERE AUTHORED:
Initially, the 2017 Edition was "compiled" based upon the 2011 Edition of the SDK. Now, the SDK is "compiled" entirely using an earlier baseline revision. Sources for this compiler are written in C# and the .Net framework on Windows and are available on github. POS bits are mostly acquired using NLTK (http://www.nltk.org) via TextBlob (http://textblob.readthedocs.io/en/dev/index.html) which itself is wrapped in a custom Django Python HTTP service end-point (this also is available on github). The SDK "compiler" calls the Django REST service for POS markings and sentence segmentation verse-by-verse. POS bits are augmented to capture case markings on pronouns and verb forms.

Digital AV –Record layouts & File inventory

2018 Edition – SDK Revision: I-728

LICENSE REQUIREMENT:

- In order to comply with the MIT-style open source license, please include AV-License.txt with your distribution of any file identified in this SDK

RECENT RESOLUTIONS & CAVEATS:

- Fundamental SDK format has stabilized and is not expected to change at all for the 2017 releases.
- As of the #HA29 release, Part-of-speech (POS) bits in AV-Writ.DX3 & AV-Writ.DX8 are new. Transition bits have also been updated in all three AV-Writ.* files. Additionally, the #HC11 revision adds a bit to detect the word on which to place a close parenthesis. However, the current AV-Writ.ascii does not reflect this new bit. So while the AV-Writ.DX* files are fully revisioned to #HC11, AV-Writ.ascii is only revisioned to #HA29. To be clear, the only difference between #HA29 and #HC11 is the absence of the close-parenthis 0x8 bit in #HA29.

RELEASE NOTES:

- #1 Revision numbers use a four-digit character sequence. The character sequence can be interpreted as Y-MDD and it identifies the date of the release. Year (Y) is encoded as a single base-36 digit: (Y = 0) represents 2000; (Y = 9) represents 2009; (Y = A) represents 2010; (Y = H) represents 2017. With respect to months, digits 1 through 9 are as expected; A is October; B is November; and C is December. DD is simply a two-digit decimal number between 01 and 31.
- #3 Multiple revision numbers exist. The Digital-AV SDK revision (aka, the "plate" revision) is the most significant set of files; it applies to all files that are prefixed with AV-. AVX-extensions are optional modules and are prefixed with AVX-, not that all files in the SDK are required to produce working bible software. The sample source code provided in avtext.go depicts a fairly minimal set of file usage while still providing access to the entire AV Bible text (none of the AVX- files are referenced by the avtext.go sample source code).
- #4 For several of the binary files, there exist also corresponding text files with a .ascii extent. These files are not provided for runtime execution. Instead, they should be considered as ancillary documentation to shed light on the corresponding binary files.
- #5 Document revisions can occur without any corresponding changes to other SDK files (Sometimes, we just find a better way of explaining what is in the SDK).

DIRECT LINKS TO THE DIGITAL-AV, AVX, AND SDK SAMPLE SOURCE FILES:

This document	http://avtext.org/SDK/Digital-AV.pdf
AV-License.txt	http://avtext.org/SDK/AV-License.txt
AV-Writ.DX2	http://avtext.org/SDK/AV-Writ.DX2
AV-Writ.DX3	http://avtext.org/SDK/AV-Writ.DX3
AV-Writ.DX8	http://avtext.org/SDK/AV-Writ.DX8
AV-Writ.ascii	http://avtext.org/SDK/AV-Writ.ascii
AV-Book.IX4	http://avtext.org/SDK/AV-Book.IX4
AV-Chapter.IX4	http://avtext.org/SDK/AV-Chapter.IX4
AV-Verse.IX2	http://avtext.org/SDK/AV-Verse.IX2
AV-Lexicon.VLT	http://avtext.org/SDK/AV-Lexicon.VLT
AVX-Lexicon.VLT	http://avtext.org/SDK/AVX-Lexicon.VLT
AVX-Lexicon.VLT	http://avtext.org/SDK/AVX-Lexicon.DB
AV-Lemma.VLT	http://avtext.org/SDK/AV-Lemma.VLT
AV-Inventory.bom	http://avtext.org/SDK/AV-Inventory.bom
avtext.go	http://avtext.org/SDK/avtext.go
AV-Stylesheet.css	http://avtext.org/SDK/AV-Stylesheet.css

Sources and Plates now on github.com:

https://github.com/kwonus/avx