A Dynamic Storage Model For Assyrian Computer Text
Peter (Jasim) BetBasoo
Nineveh Software Corporation

*off-print from*

SyrCOM-95

Proceedings of the First International Forum On Syriac Computing
(In Association wit Syriac Symposium II)

June 8, 1995
The Catholic University of America
Washington, D.C.

Edited by
George Anton Kiraz
University of Cambridge
(St. John's College)

Published by the Syriac Computing Institute
A Dynamic Storage Model for Assyrian Computer Text

Peter Jasim
Nineveh Software Corporation

In this paper I shall propose a model of representing Assyrian text in computer memory, and discuss proposed standards for an Assyrian keyboard and data interchange code. I shall also discuss the recent developments in the Unicode standard.

In his paper On the Design of an Assyrian Word Processing System (JAAS, Volume V, No. 2), Sargon Hasso proposes what I shall term a Static Storage Model (SSM) for representation of Assyrian text in a computer. The fundamental properties of SSM are:

A glyph object structure composed of a character and a diacritical mark is used to represent a glyph (character+diacritical mark). A character requires 1 byte of storage, as does a diacritical mark; the minimum storage for a glyph is, therefore, 2 bytes. The glyph object structure can be visualized as follows:

Character::Diacritical mark

A lookup table is used to render each glyph. This implies that all possible combinations of characters and diacritical markings have been defined and placed in this lookup table. It is for this reason that I call this the Static Storage Model.

In the Static Storage Model there is a many-to-one relationship between what is internally stored in the computer and what is rendered on an output device (such as a monitor or printer). For example, the following

\[ \text{i} \]

is stored internally as 65:97 (glyph codes are defined in Appendix A). A computer would use these codes to find the predefined glyph Alap+Zqapa in a lookup table. Each glyph object will have a unique entry in the lookup table. Here is an example for the word 𒍎}

<table>
<thead>
<tr>
<th>Glyph</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>66:98</td>
<td>(\text{i})</td>
</tr>
<tr>
<td>74:0</td>
<td>(\text{ })</td>
</tr>
<tr>
<td>86:97</td>
<td>(\text{ })</td>
</tr>
<tr>
<td>65:0</td>
<td>(\text{ })</td>
</tr>
</tbody>
</table>

Zero indicates no diacritical mark. Eight bytes are used to represent this word, two of which (the zeroes) are unused. It is important to realize that under SSM the computer has every possible glyph predefined in the lookup table. For this reason, the computer cannot represent any new combination of character and diacritical mark. Assuming there are 22 letters and about 20 diacritical marks, the lookup table would contain at least 22*20, or 440 glyphs. This assumes that a letter can have only one diacritical, which is not the case; the actual size of the lookup table will, therefore, be larger.

Another limitation of SSM is that it cannot represent multiple diacritical marks on the same character in an efficient way. For example, in the word 𒍎𒍎, Gamal has two diacritical marks. The glyph object structure, however, can only store one. SSM fails in
this case. This problem can be solved by special processing, but this comes at the expense of generality and complex algorithms.

In SSM the glyph object structure is a character and a diacritical mark. This leads to unusual and undesirable editing operations. If a user presses the delete key, what should be deleted, the character or the diacritical mark? Separate keys must be used to delete characters and diacritical marks.

To summarize, the Static Storage Model makes inefficient use of memory, and it cannot handle characters with multiple diacritical marks. SSM also has a many-to-one relationship between internal storage and external representation, which forces the development of very complex rendering algorithms. In addition, many unusual and undesirable effects arise, all because of a poorly designed data structure. There is a far simpler alternative to SSM.

A Dynamic Storage Model

The Dynamic Storage Model (DSM) has the following fundamental properties:

Each letter or diacritical mark is stored as a unique, 1 byte code, separately and independently of its neighbors.

Each character or diacritical mark has a location property, which tells the computer where it should be placed: at the previous position, at the current position, or at the next position.

Each character or diacritical mark has a cursor effect property, which tells the computer how to move the cursor: backward, no motion, or forward.

A lookup table, which is called a font, is defined to contain only atomic glyphs; i.e., individual characters and diacritical marks. The computer dynamically combines these to produce various combinations of characters and diacritical marks. The font will contain, at most, 223 glyphs.

The Dynamic Storage Model has a glyph object structure which is 1 byte in length. Here is the previous example using DSM

<table>
<thead>
<tr>
<th>Glyph</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>66</td>
<td>a</td>
</tr>
<tr>
<td>98</td>
<td>:</td>
</tr>
<tr>
<td>74</td>
<td>,</td>
</tr>
<tr>
<td>86</td>
<td>&gt;</td>
</tr>
<tr>
<td>97</td>
<td>-</td>
</tr>
<tr>
<td>65</td>
<td>1</td>
</tr>
</tbody>
</table>

The following properties are true of DSM

1. DSM requires less storage space. Only six bytes are required to store this word, whereas SSM requires eight bytes -- a 25% reduction in storage space.
2. Each glyph is stored consecutively in memory.
3. There is a one-to-one relationship between internal and external representation.
4. The diacritical marks Ptakha and Zqapa have the following properties
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

<table>
<thead>
<tr>
<th>Location</th>
<th>Property</th>
<th>Cursor Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Piakha</td>
<td>previous position</td>
<td>no motion</td>
</tr>
<tr>
<td>Zagapa</td>
<td>previous position</td>
<td>no motion</td>
</tr>
</tbody>
</table>

The remaining diacritical marks are similarly defined (Appendix E).

DSM handles multiple diacritical marks on the same letter in a natural and intuitive way. For example, the word :System: is stored as follows:

<table>
<thead>
<tr>
<th>Glyph</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>85</td>
<td>𓄩</td>
</tr>
<tr>
<td>70</td>
<td>𓄖</td>
</tr>
<tr>
<td>103</td>
<td>𓄲</td>
</tr>
<tr>
<td>67</td>
<td>𓄵</td>
</tr>
<tr>
<td>103</td>
<td>𓄲</td>
</tr>
<tr>
<td>110</td>
<td>𓄲</td>
</tr>
<tr>
<td>76</td>
<td>𓄵</td>
</tr>
<tr>
<td>97</td>
<td>𓄲</td>
</tr>
<tr>
<td>65</td>
<td>𓄲</td>
</tr>
</tbody>
</table>

DSM does not impose unusual editing operations on the user. For example, a delete operation would delete the glyph currently pointed to, be it a character or a diacritical mark. Hence, one key would be used for deletion, thus maintaining complete generality.

I have touched upon only a few of DSM’s properties. There are many technical issues which arise in implementing DSM in a software system; it is beyond the scope of this paper to discuss these in detail. Appendix E contains the DSM specification for Eastern Assyrian. As can be seen from Appendix E, there is very little, aside from the script, that is specific to Eastern Assyrian (and not to Western Assyrian or Estrangelo). DSM transparently handles all three cases.

**Four Essential Standards**

Uniform standards are crucial for the development of hardware and software systems. The two most basic standards are a standard keyboard layout and a standard data interchange code, as well as a font standard and a contextual analysis standard. These four standards work conjunctively; it is not possible to omit one without effecting the system.

**Data Interchange Code**

A Data Interchange Code allows one computer to communicate with another. For example, it would be undesirable to have one computer store the letter Alap as 65, and another to store it as 100. Documents written on one machine would display garbage when shown on the second. In addition, a standard code is necessary for proper lexical operations, such as searching and sorting. Once again, I present the standard that was developed at the First Ashurbanipal Library Computer Conference, but slightly modified for improvement. This standard is called SACII, Standard Assyrian Code for Information Interchange. Please refer to Appendix A.
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

Keyboard Layout

It is important to have a standard Assyrian keyboard layout so that, once having learned the layout, a person can sit and use any Assyrian keyboard without retraining. The Assyrian Standard Keyboard Layout (ASKL) was developed at the First Ashurbanipal Library Computer Conference. The layout is based on a computer analysis of the frequency of use of each Assyrian letter. The most often used letters are placed near the center of the keyboard, and the least used are placed to either side (refer to the Proceedings of the First Ashurbanipal Library Computer Conference for more details). I have modified ASKL slightly since the original standard was published, mainly to make it compatible with modern operating systems (i.e., OS/2, Windows, Macintosh), and to remove the reliance on special shift keys. ASKL is shown in Appendix B.

Contextual Analysis

It is not possible to have a practical keyboard layout standard without contextual analysis, since letters in the Assyrian alphabet change shape depending on their position in a word. Appendix C specifies a standard method of contextual analysis.

Font Standard

Every Assyrian font, be it Eastern, Western, Estrangelo, or a new, modern creation, must conform to the font standard prescribed in Appendix D. The font standard is a corollary of SACII, and it is stated explicitly for emphasis.

Application of the model

Appendix E contains a specification for the Eastern Assyrian font based on the concepts developed in this paper. As can be seen, the combination of DSM and the proposed standards provides a robust approach to the problem of computerizing the Assyrian language.

Unicode and the Assyrian Language

There are two prevailing standards for information interchange codes, ASCII (American Standard Code for Information Change), which is used by all personal computers, and EBCDIC (Extended Binary Coded Data Interchange Code), which is used mainly by IBM mainframe computers. Both ASCII and EBCDIC define 256 codes for data interchange. For example, in ASCII the letter A is code 65, the letter B is code 66, and so on. Because ASCII and EBCDIC are limited to 256 codes, they cannot handle a language that has more than 256 characters (such as Japanese). Unicode was developed to solve this problem; it provides 65,536 codes for use, which is enough to encode all of the world’s languages. Unicode will, it is pleasing to know, support Assyrian as well. The author and Sargon Hasso have submitted the Assyrian Unicode Standard to the Unicode Consortium, which has accepted the Assyrian Standard and is in the process of ratifying it.

Conclusions

In this paper I have presented a powerful storage model for representing Assyrian text in computer memory. I have also proposed standards for keyboard layout and data interchange codes. It is important to understand that DSM, ASKL, and SACII are dialect independent, i.e., they work with Eastern Assyrian, Western Assyrian (Serto), and Estrangelo. Indeed, if a computer system implements DSM and the proposed standards, a user will be able to switch from one font (Eastern, Western, or Estrangelo) to another at will, or to convert text written
in one font to another with one hundred percent accuracy, or to type text in any font in a uniform way.

References


DeKelaita, Joseph. *Grammar of the Aramaic Language*. Assyrian Church of the East Press. 1929


Appendix A

SACII
Standard Assyrian Code for Information Interchange

This appendix lists the Standard Assyrian Code for Information Interchange (SACII), from 0 to 255.

To ensure that data can be transferred from one computer to another, all vendors should use SACII faithfully. SACII provides a broad, fundamental foundation upon which all applications will be developed, such as database management, lexical analysis, spelling checkers, and so forth.

SACII defines the letters, symbols, and punctuation marks used in writing Assyrian. This includes Eastern Assyrian, Western Assyrian (Serto), and Estrangelo. As a matter of fact, no distinction is made between these three fonts at this level.

SACII fully supports contextual analysis. It reserves codes for the free, initial, middle, and final forms of each letter.

In the following table each letter is shown in Eastern, Estrangelo, and Western, in that order.

<table>
<thead>
<tr>
<th>Code</th>
<th>Symbol</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-31</td>
<td></td>
<td>same as ASCII control codes</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>Shopa speeqa (2340)</td>
</tr>
<tr>
<td>33</td>
<td>!</td>
<td>Neoshanga dpoqdana (2340)</td>
</tr>
<tr>
<td>34</td>
<td>&quot;</td>
<td>Munrana (2340)</td>
</tr>
<tr>
<td>35</td>
<td>#</td>
<td>Minyana (2340)</td>
</tr>
<tr>
<td>36</td>
<td>&quot;</td>
<td>Rahta (2340)</td>
</tr>
<tr>
<td>37</td>
<td>%</td>
<td>Immooona (2340)</td>
</tr>
<tr>
<td>38</td>
<td>:</td>
<td>'aseer (2340)</td>
</tr>
<tr>
<td>39</td>
<td>'</td>
<td>Mkhoyiddana (2340)</td>
</tr>
</tbody>
</table>
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

40 ( Qishta simmaluita ( مصدر اسمالمد )
41 ) Qishta yameenaita ( مصدر اسميند )
42 : Miakspana ( مصدر اسمضن )
43 + Mazyiddana ( مصدر اسميد )
44 , Neeshanka dnoohara ( مصدر اسمحلا )
45 - Mapserana ( مصدر اسمز )
46 . Pasoga ( مصدر اسم الأ )
47 / Palee'ana ( مصدر اسم الب )
48 0 Seepar ( مصدر اسم ب )
49 1 Kha ( مصدر اسم خ )
50 2 Tre ( مصدر اسم ث )
51 3 Tlata ( مصدر اسم ثلا )
52 4 Arb'a ( مصدر اسم أرب )
53 5 Khamsha ( مصدر اسم خمسم )
54 6 Ishxa ( مصدر اسم إيش )
55 7 Shaw'a ( مصدر اسم ش )
56 8 Timanya ( مصدر اسم ثماس )
57 9 Tish'a ( مصدر اسم تيساس )
58 : Zawg'a ( مصدر اسم ضؤ )
59 : Pasoga kirya ( مصدر اسم ضؤ ب )
60 < Soora min ( مصدر اسم سوس )
61 = Dma ( مصدر اسم دما )
62 > Goora min ( مصدر اسم قوس )
63 ? Neeshanka dshooala ( مصدر اسم سوشلا )
64 . Napsa ( مصدر اسم ناس )
65 1 ( ) Allap
66 0 ( ) Bei
67 ( ) Gamnal
68 0 ( ) Dallai
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

69  א (א א) Nel
70  ב (ב ב) Wow
71  ג (ג ג) Zen
72  ד (ד ד) Kheth
73  ה (ה ה) Teth
74  י (י י) Yood
75  ק (ק ק) Kap
76  ל (ל ל) Lammad
77  מ (מ מ) Meem
78  נ (נ נ) Noon
79  ס (ס ס) Sin Ket
80  ת (ת ת) 'e
81  ע (ע ע) Pe
82  פ (פ פ) Sade
83  ס (ס ס) Qop
84  צ (צ צ) Resh
85  ק (ק ק) Sheen
86  ה (ה ה) Tow
87  ะ (ะ ะ) reserved for 23rd Mandaic letter
88  מז'נה (מזהנה) Mz'ana
89  ריעה (ריעה) Rima
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

90  
Msha’lana (MEDM)

91  
Samka (MED)

92  
Mnakhia (MED)

93  
Rahua dkarte (MED)

94  
Marmana (MED)

95  
Serta dkhooyada (MED)

96  
Zqapa (MED, Western)

97  
Pikaka (MED, Western)

98  
Zlame psheeqe (MED)

99  
Zlame qishye (MED, Western)

100  
Khvasa (MED, Western)

101  
Rwakha (MED, Western)

102  
Rwasa (MED, in Eastern and Western)

103  
Syana (MED)

104  
Rimkha (MED, below letter)

105  
Qishta (MED, below letter)

106  
Majleeyana (MED, below letter)

107  
Serta khvera (MED, below letter)

108  
Serta ‘elayta (MED, above letter)

109  
Tulqana (MED, back 02, for silent or accented letters)

110  
‘elaye (MED, rests on either side of letter)

111  
Takhtaye (MED, rests on either side of letter)

112  
Kikhwa (MED)

113  
Slecwa (MED)

114  
Mhagyan 2 (MED, below letter)

115  
Mhagyan (MED, below letter)

116  
Gneez (MED)

117  
Stoona goonya (MED)
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

119  Stoon a khitaya ( $$$
120  Stoon a 'eluya ( $$$
121  left bracket
122  right bracket
123  unused
124  Toopa raa saltaya ( $$$
125  Toopa raa yaneenaya ( $$$
126  start rendering Sharee makhma ( $$$
127  stop rendering Maklee makhma ( $$$
128 - 150 reserved for free forms
151 - 173 reserved for initial forms
174 - 196 reserved for final forms
197 - 255 font specific (such as ligatures)

Codes 101 and 103 require clarification. In Eastern Assyrian ꜌ = ee (as in sheet), in
Western Assyrian it is the diacritical mark ꜌ which is ee. As far as the computer is
concerned, ꜌ is just a letter with a dot under it, as are ꜌ ꜌ ꜌. One key, therefore, serves
all these purposes. However, when one switches to a Western font the ꜌ becomes ꜌,
and there no longer is a key for ꜌, which is still needed. Two codes must be defined,
therefore, to guarantee a one-to-one relationship between Eastern and Western text. Code
101 has the same meaning in both Eastern and Western (ee); code 103 always means a dot
under a letter, regardless of the font.

Codes 126 and 127 instruct the software to control rendering. This is useful in cases
where a ligature, such as ꜌ , is not desired -- code 127 would force the software to show
 ꜌ in this case.
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

Appendix B

ASKL
Assyrian Standard Keyboard Layout

This appendix lists the complete definition of the Assyrian Standard Keyboard Layout (ASKL). ASKL was designed based on the frequency of use of each Assyrian letter; the most frequently used letters are placed in the center of the keyboard, and the letters least frequently used are placed on either side.

The following specification assumes contextual analysis (appendix C); keys are listed from top row to bottom row, from left to right as seen on a QWERTY (English) keyboard. The following key combinations are defined in SSKL.

<table>
<thead>
<tr>
<th>QWERTY Key</th>
<th>ASKL</th>
<th>SSSI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>49</td>
<td>Kha (/spi/)</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>50</td>
<td>Tre (tepem)</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>51</td>
<td>Tinta (tala)</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>52</td>
<td>Arb'a (بارب)</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>53</td>
<td>Khamsha (kamza)</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>54</td>
<td>Ish'a (ishma)</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>55</td>
<td>Shaw'a (shama)</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>56</td>
<td>Trimya (trima)</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>57</td>
<td>Tish'a (tisma)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>48</td>
<td>Seepur (seput)</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>45</td>
<td>Mapserana (mapserana)</td>
</tr>
<tr>
<td>=</td>
<td>=</td>
<td>61</td>
<td>Dma (dama)</td>
</tr>
<tr>
<td>Q</td>
<td>81</td>
<td>83</td>
<td>Qop</td>
</tr>
<tr>
<td>W</td>
<td>82</td>
<td>80</td>
<td>'e</td>
</tr>
<tr>
<td>E</td>
<td>83</td>
<td>101</td>
<td>Khwasa (khuza)</td>
</tr>
<tr>
<td>R</td>
<td>100</td>
<td>100</td>
<td>Zlame qishye (zlama)</td>
</tr>
</tbody>
</table>
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

<table>
<thead>
<tr>
<th>Letter</th>
<th>Code</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>98</td>
<td>Pinakha (𒆠𒆜 Western)</td>
</tr>
<tr>
<td>Y</td>
<td>97</td>
<td>Zqapa (𒆠𒆜, Western)</td>
</tr>
<tr>
<td>U</td>
<td>99</td>
<td>Zlame pshege (𒆠𒆜)</td>
</tr>
<tr>
<td>I</td>
<td>102</td>
<td>Rwakha (𒆠𒆢 Western)</td>
</tr>
<tr>
<td>O</td>
<td>72</td>
<td>Kheth</td>
</tr>
<tr>
<td>P</td>
<td>81</td>
<td>Pe</td>
</tr>
<tr>
<td>I</td>
<td>67</td>
<td>Gammal</td>
</tr>
<tr>
<td>J</td>
<td>82</td>
<td>Sade</td>
</tr>
<tr>
<td>A</td>
<td>79</td>
<td>Simket</td>
</tr>
<tr>
<td>S</td>
<td>69</td>
<td>Hea</td>
</tr>
<tr>
<td>D</td>
<td>68</td>
<td>Dollau</td>
</tr>
<tr>
<td>F</td>
<td>70</td>
<td>Wow</td>
</tr>
<tr>
<td>G</td>
<td>74</td>
<td>Yood</td>
</tr>
<tr>
<td>H</td>
<td>78</td>
<td>Nuon</td>
</tr>
<tr>
<td>J</td>
<td>65</td>
<td>Allap</td>
</tr>
<tr>
<td>K</td>
<td>77</td>
<td>Meem</td>
</tr>
<tr>
<td>L</td>
<td>76</td>
<td>Lannad</td>
</tr>
<tr>
<td>:</td>
<td>59</td>
<td>Pasaqa kirya (𒆠𒆠)</td>
</tr>
<tr>
<td>,</td>
<td>39</td>
<td>Mkhayiddana (𒆠)</td>
</tr>
<tr>
<td>Z</td>
<td>71</td>
<td>Zen</td>
</tr>
<tr>
<td>X</td>
<td>73</td>
<td>Teth</td>
</tr>
<tr>
<td>C</td>
<td>75</td>
<td>Kap</td>
</tr>
<tr>
<td>V</td>
<td>86</td>
<td>Tow</td>
</tr>
<tr>
<td>B</td>
<td>66</td>
<td>Bet</td>
</tr>
<tr>
<td>N</td>
<td>84</td>
<td>Resh</td>
</tr>
<tr>
<td>M</td>
<td>85</td>
<td>Sheen</td>
</tr>
<tr>
<td>.</td>
<td>44</td>
<td>Neeshanga dnuohara (𒆠𒆠)</td>
</tr>
<tr>
<td>,</td>
<td>46</td>
<td>Pasaqa (𒆠)</td>
</tr>
<tr>
<td>I</td>
<td>47</td>
<td>Palee'ana (𒆠)</td>
</tr>
<tr>
<td>SHIFT 1</td>
<td>33</td>
<td>Neeshanga dpoogdana (𒆠)</td>
</tr>
<tr>
<td>SHIFT 2</td>
<td>36</td>
<td>Rahia (𒆠)</td>
</tr>
<tr>
<td>Shift</td>
<td>Key</td>
<td>Value</td>
</tr>
<tr>
<td>-------</td>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>Shift 3</td>
<td>35</td>
<td>Minyana</td>
</tr>
<tr>
<td>Shift 4</td>
<td>64</td>
<td>Napsa</td>
</tr>
<tr>
<td>Shift 5</td>
<td>37</td>
<td>Immoona</td>
</tr>
<tr>
<td>Shift 6</td>
<td>94</td>
<td>Marmana</td>
</tr>
<tr>
<td>Shift 7</td>
<td>38</td>
<td>'aseer</td>
</tr>
<tr>
<td>Shift 8</td>
<td>42</td>
<td>M'takpana</td>
</tr>
<tr>
<td>Shift 9</td>
<td>40</td>
<td>Qishta simmalaita</td>
</tr>
<tr>
<td>Shift 0</td>
<td>41</td>
<td>Qishta yameenaita</td>
</tr>
<tr>
<td>Shift -</td>
<td>95</td>
<td>هذپا</td>
</tr>
<tr>
<td>Shift =</td>
<td>43</td>
<td>Maryiddana</td>
</tr>
<tr>
<td>Shift Q</td>
<td>96</td>
<td>Stoona goonya</td>
</tr>
<tr>
<td>Shift W</td>
<td>118</td>
<td>Talqana</td>
</tr>
<tr>
<td>Shift E</td>
<td>110</td>
<td>for silent or accented letters</td>
</tr>
<tr>
<td>Shift R</td>
<td>111</td>
<td>'etaye</td>
</tr>
<tr>
<td>Shift T</td>
<td>113</td>
<td>Kakhwa</td>
</tr>
<tr>
<td>Shift Y</td>
<td>112</td>
<td>Takhuaye</td>
</tr>
<tr>
<td>Shift U</td>
<td>104</td>
<td>Syame</td>
</tr>
<tr>
<td>Shift I</td>
<td>107</td>
<td>Majleeyana</td>
</tr>
<tr>
<td>Shift O</td>
<td>120</td>
<td>Stoona 'elaya</td>
</tr>
<tr>
<td>Shift P</td>
<td>119</td>
<td>Stoona khtaya</td>
</tr>
<tr>
<td>Shift [</td>
<td>121</td>
<td>left bracket</td>
</tr>
<tr>
<td>Shift ]</td>
<td>122</td>
<td>right bracket</td>
</tr>
<tr>
<td>Shift A</td>
<td>108</td>
<td>Serta khetta</td>
</tr>
<tr>
<td>Shift S</td>
<td>109</td>
<td>Serta 'elaya</td>
</tr>
<tr>
<td>Shift D</td>
<td>106</td>
<td>Qishta</td>
</tr>
<tr>
<td>Shift F</td>
<td>105</td>
<td>Rinkha</td>
</tr>
<tr>
<td>Shift G</td>
<td>114</td>
<td>Sleewa</td>
</tr>
<tr>
<td>Shift H</td>
<td>103</td>
<td>Rwasa</td>
</tr>
<tr>
<td>Shift J</td>
<td>115</td>
<td>Mhagyana 2</td>
</tr>
<tr>
<td>Shift K</td>
<td>116</td>
<td>Mhagyana</td>
</tr>
</tbody>
</table>
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

SHIFT L ... 117 Gneez (گنیز)
SHIFT ; : 58 Zawga (ژاوگا)
SHIFT ' " 34 Mannana (مننانا)
SHIFT Z . 88 Mzi'ana (مژیانا)
SHIFT X . 89 Riima (ریما)
SHIFT C 90 Msha'iana (مسحیانه)
SHIFT V . 91 Sanika (سنیکه)
SHIFT B . 92 Mnakhta (منکتکه)
SHIFT N
SHIFT M . 93 Rahta dakrie (رحت دکریه)
SHIFT , < 60 Soora min ( sứری مین)
SHIFT , > 62 Goora min (جوری مین)
SHIFT / \ 63 Neeshanga dshoalo (نیشنگانه دشوالو)

ASKL
Assyrian Standard Keyboard Layout

Note the following equivalences

<table>
<thead>
<tr>
<th>Eastern</th>
<th>Western</th>
</tr>
</thead>
<tbody>
<tr>
<td>١٥٠٥</td>
<td>١٥٠٥</td>
</tr>
<tr>
<td>١٥٠٥</td>
<td>١٥٠٥</td>
</tr>
<tr>
<td>١٥٠٥</td>
<td>١٥٠٥</td>
</tr>
<tr>
<td>١٥٠٥</td>
<td>١٥٠٥</td>
</tr>
<tr>
<td>١٥٠٥</td>
<td>١٥٠٥</td>
</tr>
<tr>
<td>١٥٠٥</td>
<td>١٥٠٥</td>
</tr>
</tbody>
</table>

94
Appendix C

Contextual Analysis

Contextual Analysis is a development made possible by the computer. Very simply put, Contextual Analysis is the ability of the computer to automatically place the correct shape of a letter into a word. For example, the word 𒃜𒃜 requires the following Contextual Analysis:

space 𒃜𒃜 Key pressed
space 𒃝𒃝 Computer shows

After pressing space, the computer changes the final 𒃜 in the word to 𒃝. Thus all a typist needs to type is one letter and the computer determines which shape of that letter to place in the word; this means that there would be only 22 letter keys on the Assyrian keyboard.

The following is a basic algorithm for contextual analysis. Note, this algorithm does not include support for font-specific rendering (see Appendix E).

Step 1 Get keystroke
Is it a space?
   Yes (a space was typed)
      Beginning of the document?
         No (not beginning of document)
            Is previous character a letter?
               Yes (it's a letter)
                  Is letter preceded by a space?
                     Yes (preceded by a space)
                        Put free form
                     No (not preceded by a space)
                        Change it to final form
                  Put space (type the keystroke)
               No (something other than space was typed)
                  Is keystroke a letter?
                     Yes
                        Beginning of the document or previous character a space?
                           Yes (beginning of document or space)
                              Put initial form
                           No (not beginning of document or space)
                              Put middle form
                     No
                        Type the keystroke
               Go to step 1
Appendix D

Font Standard

An Assyrian font must define, at a minimum, the character set defined by SACII (Appendix A, codes 32-196). While the shape of each character will differ from font to font, the identity of the character will remain the same.

Every Assyrian font must have four forms for each letter:

1. **free** letter is not connected on either side.
2. **initial** letter is not connected on right side and is connected on the left side.
3. **middle** letter is connected on both sides.
4. **final** letter is connected on right side and is not connected on the left side.

Contextual Analysis will automatically place the correct form of the letter into the word. SACII defines the following codes for each of these forms:

<table>
<thead>
<tr>
<th>Form</th>
<th>SACII code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Middle forms</td>
<td>65 - 87</td>
</tr>
<tr>
<td>Free forms</td>
<td>128 - 150</td>
</tr>
<tr>
<td>Initial forms</td>
<td>151 - 173</td>
</tr>
<tr>
<td>Final forms</td>
<td>174 - 196</td>
</tr>
</tbody>
</table>
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

Appendix E
DSM Specification for Eastern Assyrian

This appendix uses the dynamic storage model and the standards developed in Appendix A, Appendix C, and Appendix D to define the properties of the Eastern Assyrian font. The following properties are defined:

P1. The four shapes of each letter
P2. Connection property of each character
P3. Location property of each character
P4. Cursor effect property of each character
P5. Ligatures
P6. Rendering rules

Contextual analysis is font specific. Eastern Assyrian, Western Assyrian, and Estrangelo require different rules of rendering and different ligatures (P5 and P6). SACII supports contextual analysis by reserving codes for the free, initial, middle, and final forms of each letter (P1 and P2). These codes provide a standard, font independent method of rendering the three major Assyrian fonts. There are, however, differences in the fonts which are not encoded in SACII, and which must be handled algorithmically. These rendering rules (P5 and P6) must be specified for each Assyrian font.

The following is the specification for Eastern Assyrian. Specifications for Western Assyrian and Estrangelo remain to be developed.

The following table defines the first five properties, P1-P5, of Eastern Assyrian.

<table>
<thead>
<tr>
<th>SACII</th>
<th>Symbol</th>
<th>Connections Left, Right</th>
<th>Location Property Previous, Current</th>
<th>Cursor Effect Backward, None</th>
</tr>
</thead>
<tbody>
<tr>
<td>32-35</td>
<td>N</td>
<td>N</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>36</td>
<td>.</td>
<td>N</td>
<td>P</td>
<td>N</td>
</tr>
<tr>
<td>37</td>
<td>%</td>
<td>N</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>38</td>
<td>;</td>
<td>N</td>
<td>P</td>
<td>N</td>
</tr>
<tr>
<td>39-41</td>
<td>N</td>
<td>N</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>42</td>
<td>:</td>
<td>N</td>
<td>P</td>
<td>N</td>
</tr>
<tr>
<td>43-63</td>
<td>N</td>
<td>N</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>64</td>
<td>.</td>
<td>N</td>
<td>P</td>
<td>N</td>
</tr>
<tr>
<td>65</td>
<td>R</td>
<td>R</td>
<td>C</td>
<td>F (middle forms, 65-87)</td>
</tr>
<tr>
<td>66</td>
<td>RL</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>67</td>
<td>RL</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>68</td>
<td>R</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>69</td>
<td>R</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>70</td>
<td>R</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>71</td>
<td>R</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>72</td>
<td>RL</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>73</td>
<td>RL</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>74</td>
<td>RL</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>75</td>
<td>RL</td>
<td>R</td>
<td>C</td>
<td>F</td>
</tr>
</tbody>
</table>
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

121  
122  
123 unused
124  
125  
126-127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150 reserved
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  

(Free forms, 128-150)

(Initial forms, 151-173)
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

165       L       P       N
166       L       P       N
167       L       P       N
168       L       P       N
169       L       P       N
170       L       P       N
171       L       P       N
172       L       P       N
173       reserved
174       R       P       N (final forms, 174-196)
175       R       P       N
176       R       P       N
177       R       P       N
178       R       P       N
179       R       P       N
180       R       P       N
181       R       P       N
182       R       P       N
183       R       P       N
184       R       P       N
185       R       P       N
186       R       P       N
187       R       P       N
188       R       P       N
189       R       P       N
190       R       P       N
191       R       P       N
192       R       P       N
193       R       P       N
194       R       P       N
195       R       P       N
196       reserved
197       ligature for a
198       ligature for ax
199       ligature for ax
200       ligature for ox
201-255   unused
A Dynamic Storage Model For Assyrian Computer Text
Peter BetBasoo

Rendering rules (P5, P6)

A word begins with a space and ends with a space, and cannot contain a space.

If a user inserts the suspend rendering code (127) before a character then that character is printed as is, without special rendering -- the following rules would not apply.

Rules of rendering

R1 Left and right tails are attached to a letter that is preceded or followed by a space, or both. Letters which accept a right tail are: ɐ ɐ ɐ. Letters which accept a left tail are:

R2 If ɐ appears at the end of a word and is not preceded by ɐ or ɐ, then ɐ is replaced by ɐ.

R3 If ɐ appears at the end of a word and is preceded by a letter that does not connect on its left, then ɐ is replaced by ɐ.

R4 If ɐ appears at the end of a word and is preceded by a letter that connects on its left, then ɐ is replaced by ɐ.

R5 If ɐ appears at the end of a word and is preceded by a letter that does not connect on its left, then ɐ is replaced by ɐ.

R6 If ɐ appears at the end of a word and is preceded by a letter that connects on its left, then ɐ is replaced by ɐ.

R7 If ɐ ɐ appear at the end of a word then ɐ ɐ are replaced by the ligature ɐ or ɐ, depending on a default set by the user.

R8 If the word ɐ ɐ appears then it is replaced by the ligature ɐ ɐ.