附加说明

V1.8(2021.3.8)

一、本转换规则适用范围

目前，本转换规则只包含现代蒙古文名义字符及其变形显现字符。在设计上基于蒙古文编码国家标准 GB25914-2010，并对其已定义但不足之处进行了完善工作，对其未定义部分进行了补充工作。在本转换规则中未涉及的相关内容需要按照国家标准 GB25914-2010 来实施。

二、FVS（1-4）定义及 FVS4 推荐意见

自由变体选择符（FVS1，FVS2，FVS3，FVS4）在本转换规则当中的作用是在确定了词里位置（独立，词首，词中，词末）的情况下，不受其他语法上下文（如单词阴阳性，特殊字符组合等等）环境影响，语法上下文无关地强制请求当前字符的固定变体。并且字符的每个变体都固定分配了唯一的变体选择控制符，即字符的每个变体和变体选择控制符是一一对应。自由变体选择符在词首时忽略它的存在，不影响其后字符的词里位置属性。词中或词末时直接按本转换规则中定义的变形规则处理。本转换规则中未定义的情况忽略它的存在，不影响其前后字符的词里位置属性，字形不产生变化。

自由变体选择符 FVS4 是在本转换规则当中的暂时定义，该自由变体选择符在国家标准中尚未定义，拟计划新申请。在现代蒙古文的一般性需求上不会出现需要 FVS4 的情况。在实施过程中如有需要，在完成 FVS4 码位申请之前，请暂用其他适合自己实际需求的临时代替方案，如 VS01（FE00）、ZWJ（200D）、NIRUGU（180A）、FVS1+FVS1 等。本规则推荐以复合 FVS 来代替 FVS4。

推荐性替代方案如下：FVS4=FVS1+FVS1。

如[ZWJ+182D+FVS1+FVS1+ZWJ]等价于[ZWJ+182D+FVS4+ZWJ]。

三、MSC Model

MSC（Mongolian Suffix Connector） Model 主要处理蒙古文格附加成分。在本转换规则当中，穷举方式列出使用 MSC 后，使其后面字符产生字形变化的所有可能性组合。在实施过程中 MSC 等价于 MVS（用 MVS 代替 NNBSP 后）后，本节无实质性规范内容，详细参考其他章节。

四、MVS Model（MVS 字形）

MVS Model 是处理元音 1820/1821 之前分写辅音和词末分写元音 1820/1821 的变形问题。MVS 字形是为了处理不同企业的不同字体方案中，针对 MVS 之前字形的位置属性分歧（词中字形、词末字形）而提出的折中命名方式。因为有些字母的词中字形和词末字形是外形上相同的，只是命名方式不同。在本转换规则的 MVS 模型当中，虽然将其列在词中分类里，但是没有给 MVS 字形分配连贯的 FVS。即常规字形和 MVS 字形的 FVS 分配不是连贯的。目前控制符 MVS 能够使其前后字符产生字形变化的规则都以穷举方式列出。一个字符只要在 MVS 之前出现，那么无论处于独立，还是词首、词中、词末条件下，都变形为 MVS 字形。关于 MVS 在词中时阻断语法属性上下传递（蒙古文分尾元音 A/E 组合是例外）的。详细参考其它章节说明。

关于 MVS 的词里位置属性

冲突出现在连写位置模型和词里位置模型的逻辑上。它在词里使用时目前有一些模糊特性：

连写位置模型的算法逻辑上，MVS 具有断词特性，其前边字母为词末（单个字母时为独立），其后边字母为词首（单个字母时为独立）。但是在处理类似辅音 N、辅音 G、辅音 J 的词末默认字形和词中 MVS 字形有冲突时断词解释存在不好解释的现象。另外，在是否传递阴阳性等特征上有分歧（参文档[当前字母阴阳属性判断规则]）。比如 MVS 之后的 A/E 是否影响 MVS 之前的阴阳性等。正常单词里不会出现这种情况，但是善意写错和恶意写错时需要考虑有个明确的变形规则。

词里位置模型的算法逻辑上，MVS 不具有断词特性，但是，根据当前字母直连字母是普通字母、MVS、词边界等的不同而有所不同。

建议采取的方式如下：

1. 当前字母两边都是 MVS 或词边界时候，当前字母具有独立位置属性。元音 A/E 已经有了响应 MVS 的独立字形。
2. MVS 在词首时，当前字母（MVS 之后）具有词首位置属性。（当前字母后边不是 MVS 或词边界时）
3. MVS 在词末时，当前字母（MVS 之前）具有词末位置属性。在本标准中，已经给具有 MVS 字形的辅音（MVS 敏感辅音）定义了词中词末兼容的 MVS 字形。
4. MVS 在词中时：
   (1). MVS 之前字母在连写位置模型上具有词末位置属性，在词里位置模型上具有词中位置属性。无论哪种定义，对 MVS 敏感辅音（具有 MVS 字形的辅音）来说，目前已经定义了两者兼容的 MVS 字形。所以没有分歧。对其他字母（辅音和元音）一律认定为词末字形。
   (2). MVS 之后字母在连写位置模型上具有词首位置属性，在词里位置模型上具有词中位置属性。目前这个分歧比较明显，到目前未能协调成功。建议认定为词中位置属性，但是本方案认定为词首位置属性，目前暂不做评价。但是已经通过技术方式（同时定义相同属性的词中字形和词末字形）处理了这个学术分歧了。

关于 MVS 的可能的编码序列解释

简单举例 MVS 有关编码序列，及其前后字符的位置属性。

1. **一个蒙古文字符+MVS**
   - MVS 前置蒙古文字符显示为独立默认字形。
   - MVS 显示为无效图形。
2. **蒙古文字符/ZWJ/NIRUGU+一个蒙古文字符+MVS**
   - MVS 前置蒙古文字符显示为词末默认字形
   - MVS 显示为无效图形。
3. **一个蒙古文字符+MVS+A/E**
   - MVS 前置蒙古文字符显示为默认独立字形。因为 MVS 是有效的，所以是基于

语法上下文环境的默认独立字形（注意辅音 JA）。跟这个辅音是否为 MVS 敏感辅音无关。

- MVS 显示为有效图形，即元音分隔符空格（窄宽度）。
- MVS 后置 A/E 显示为分尾字形。

4. **MVS+A/E**
   - MVS 显示为有效图形，即元音分隔符（窄宽度空格）。
   - MVS 后置 A/E 显示为分尾字形。
5. **MVS+A/E+ZWJ/NIRUGU/蒙古文字母**
   - MVS 显示为有效图形，即附加成分分隔符（普通空格）。
   - MVS 后置 A/E 显示为词首字形。
6. **MVS+一个蒙古文字母（A/E 除外）**
   - MVS 显示为有效图形，即附加成分分隔符（普通空格）。
   - MVS 后置字母显示为独立默认字形。
7. **MVS+两个或两个以上蒙古文字母**
   - MVS 显示为有效图形，即附加成分分隔符（普通空格）。
   - MVS 后置字母显示为词首字形。
8. **蒙古文字符/ZWJ/NIRUGU+一个蒙古文字符+MVS+A/E**
   - MVS 前置蒙古文字符，如果其为 MVS 敏感辅音（可附带有效 FVS），那么显示为为 MVS 字形。如果其为不是 MVS 敏感辅音，那么显示为词末字形。
   - MVS 显示为有效图形，即元音分隔符（窄宽度空格）
   - MVS 后置 A/E 显示为分尾字形。
9. **一个蒙古文字符+MVS+除 A/E 以外的单个蒙古文字符**
   - MVS 前置蒙古文字符显示为独立默认字形
   - MVS 显示为有效图形，即附加成分分隔符（普通空格）
   - MVS 后置蒙古文字符显示为独立字形。
10. **一个蒙古文字符+MVS+蒙古文字符串（一个字符以上）**
    - MVS 前置蒙古文字符显示为独立默认字形
    - MVS 显示为有效图形，即附加成分分隔符（普通空格）
    - MVS 后置蒙古文字符显示为词首字形。
11. **蒙古文字符/ZWJ/NIRUGU+一个蒙古文字符+MVS+除 A/E 以外单个蒙古文字符**
    - MVS 前置蒙古文字符显示为词末字形。
    - MVS 显示为有效图形，即附加成分分隔符（普通空格）。
    - MVS 后置字符显示为独立字形。

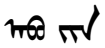
12. **蒙古文字符/ZWJ/NIRUGU+一个蒙古文字符+MVS+蒙古文字符串（一个字符以上）**

- MVS 前置蒙古文字符显示为词末字形。
- MVS 显示为有效图形，即附加成分分隔符（普通空格）。
- MVS 后置字符显示为词首字形。

五、NNBSP(U+202F)的改进方案

有的系统平台对 NNBSP（202F）的支持不够完善。比如所有苹果 iOS 系统（到现在为止）、部分 Android 系统、在微软 Office 系列中跟标点符号混合使用的时等。这一现象从蒙古文编码制定开始一致伴随至今。如果想在定制开发的系统层面解决此问题，建议采取用 MVS(U+180E)来代替 NNBSP(U+202F 的)的引导蒙古文格附加成分的功能。让 NNBSP 回归到原本[跟文种语法属性无关的窄宽度无间断空格]属性。即蒙古文 MVS 实现[蒙古文元音间隔符 MVS]和[窄宽度无间断空格 NNBSP 的引导蒙古文附加成分]的两个功能。这样又能解决部分系统平台上显示错误的问题，也能减少 NNBSP 这个控制符的使用。

我们可以将原有的 MVS 临时性重新命名为蒙古文附加成分连接符（Mongolian Suffix Connector）,简称 MSC，码位为 U+180E。附加成分里包括格附加成分和任何作为单词一部分而连在后边的后缀（暂命名为普通附加成分），包括原本被元音间隔符分离开的分尾（U+1820,U+1821）。其图形尺寸为普通空格一致。但是相应地调整分尾 a/e 的字形，如删除字形里的空白缝隙等。

文本串： 
原来的编码序列：1820+182A+1824+NNBSP+1836+1822+1828

改进的编码序列：1820+182A+1824+MSC+1836+1822+1828（MSC=U+180E）

文本串： 
原来的编码序列：182A+1820+182D+MVS+1820（MVS=U+180E）

改进的编码序列：182A+1820+182D+MSC+1820（MSC=U+180E）

※虽然这一节提到的 MSC 跟前边其他章节提到的 MSC 名称相同，但是内容和功能有所不同。上述方案暂定为临时性解决方案，只有在系统上解决目前

NNBSP 引起的问题时使用。当然后续在充分探讨论证后可以升级为国家标准和国际标准。

※不建议在蒙古文中使用 1800 编码区域以外的 NNBSP(U+202F)控制符。

六、规则举例

在本转换规则的举例包含两部分内容：一部分是现代蒙古文当中的常规词汇，另一部分是不符合现代蒙古文正字法变形规则的错误举例。

选择这些举例的原因是：

（1） 字库制作的需要：在做输入蒙古文时，因为随意输入的原因会出现“一些字符出现在不正确的位置”的情况。但是，字库里必须处理这些现象。那么必须对该字符在这个不正确的位置上定义一个字形。

（2） 人为输入一些错别字形的需要：由于研究、讲解等原因需要输入一些错别的字形。

（3） 处理一些古文献的使用方法：蒙古文中的一些拼写规则在以前使用过，但是现代蒙古文中基本不使用了。但是在某些场合还要输出这样的拼写。

（4） 转换规则尽可能转化一个字符的所有变形：由于蒙古文转换规则复杂性和发展过程的多变性，现在还不能找出每个字形的使用举例。所以尽量涵盖每个字符的所有字形的转化情况。因此，在举例里列出了一些异常的单词。

六、默认字形

在本转换规则里，每个蒙古文字符在词里的“独立、词首、词中、词末”等不同词里位置时有两种不同的默认字形的概念。语法上下文默认字形和位置属性默认字形。语法上下文默认字形是通过【词里位置属性+语法上下文环境】来计算获得。位置属性默认字形是不依赖于语法上下文，或者根本获取不到语法上下文条件情况下的默认字形，如 ZWJ+蒙古文字符+ZWJ 只能确认到上下文无关的词里词中位置属性。目前这两种默认字形都在本转换规则中明确指定。可参考转换规则内容和独立于转换规则内容的附加表格：蒙古文字符变体列表及变体选择控制符分配方案。

七、转换规则的冗余问题

如果仅从规则实施角度考虑本转换规则内容包含很多冗余规则。在指定默认字形的前提下，如果条件 A（如：当前字符在辅音之前出现）和条件 B（如：当前字符在元音之前）正好能够涵盖所有可能性，那么规则里只列一条条件 A 或条件 B，即可满足涵盖所有可能性的需求，因为其他指定条件以外的可能性自然归属到默认字形条件里。但是为了说清楚语言学正字法的所有知识点，本转换规则从语言学角度将所有可能的条件都列出来了。字体厂家在实施过程中，只要实现了本转换规则等价的转换功能，那么没有必要局限于规则的先后顺序和规则条数。

八、词里独首中末和连写独上中下的关系

在复杂语言文本当中，都存在需要明确当前字符的位置属性问题。而此位置属性是个相对的概念。如果以语言学词汇为分析问题的出发点，它就有词里的独立、词首、词中、词末等位置属性。如果以字体渲染引擎为分析问题的出发点，它就有连写的连写独、连写上、连写中、连写下等位置属性。这两种观点的主要分歧出现在元音 1820/1821 之前分写辅音和词末分写辅音的位置属性认同上。位置属性的不同认同带来的是 FVS 分配方案的不同。即【字符->词里位置属性->FVS 分配方案】和【字符->连写位置属性->FVS 分配方案】的 FVS 分配结果是明显不同的。本转换规则基于语言学词汇认知角度，采用了“词里的独立、词首、词中、词末”的位置属性分类方法，但是在 FVS 分配上采用了与连写位置属性分类法相同的分配方法。所以看 MVS 字形的 FVS 分配方案时，有明显的跳跃式分配的痕迹。

九、关于无效控制字符的提示性显示问题

以下方案为推荐性方案，字体厂家可根据自身情况进行选择性地实现。

自由变体选择符（FVS1，FVS2，FVS3，FVS4）、零宽连接符（ZWJ）、零宽禁连接符（ZWNJ）、蒙古文元音间隔符（MVS）、蒙古文附加成分连接符（MSC）在使用时是没有图形符号来显示。所以在用户使用过程中出现输入多余控制符、输入无效的控制符后发现不了的情况。如果对这些情况能够有一个提示功能，那么会避免这些情况的发生。

在字库中，应该对自由变体选择符（FVS1，FVS2，FVS3，FVS4）、零宽连接符（ZWJ）、零宽禁连接符（ZWNJ）、蒙古文元音间隔符（MVS）、蒙古文附加成分连接符（MSC）做一个对应的图形，在输入多余控制符、输入无效的控制符的情况下显示出相应的图形，给用户一个提示。

一般在下列情况下给出提示图形：

（1） 错误控制符：在输入两个或两个以上的控制符时，第二个控制符开始被认为是错误控制符。

（2） 无效控制符：给某个字符上使用没有给它分配的控制符时，应该显示出该控制符的图形。

※多余控制符是指当前字母的当前位置是分配了一个控制符，但是根据当前语法上下文，不需要控制符也能显示期望的变体。这个时候此控制符为多余控制符。多余控制符是不能显示提示图形的。

默认情况下 FVS 必须是有宽度可见格式控制字符，但在文本中有效果(FVS 影响了前边被修饰字母的显示行为)时它是不可见的。

(1). 词边界+FVS+词边界：无效（错误），显示 FVS 图形。

(2). 词边界+FVS+任意字符串：无效（错误），显示 FVS 图形。

(3). [ZWJ|MVS|FVS]{1,}+FVS+词边界：无效（错误），显示 FVS 图形。
※如果辅音 H/G 上使用 FVS+FVS 方案，这个需要另行考虑。不过这个只关系到一个字形。

(4). 蒙古文字符串+FVS+任意字符串：位置上下文环境中判断是正确使用了对应的 FVS，如使用了跟当前字母当前位置无关的 FVS。

※关于多余使用的 FVS，也能通过算法判断出，并且能够在多余使用 FVS 时显示 FVS 图形。如通过当前语法上下文能够正确选形的时候，多余地使用了当前字形对应的 FVS，那么这个叫多余使用 FVS。虽然通过显示多余 FVS 图形方式提高操作用户体验，但是因为会存在不信任文本引擎而给所有或特定字母后

边强制插入 FVS 来请求字形的应用场景。所以不建议显示多余 FVS 图形，那样容易产生过度限制。

存在问题：

替换字形方式在不同平台不同软件系统上有不同显示行为。有的系统不允许具有控制字符属性（或者其它属性）的字符不可以变形，或者永远是零宽度。如果想彻底实现此功能，可能需要修改 UCD 字符属性。通过常规处理方式，有些平台（如 word 系列）上能够按预期显示 FVS 图形，有些平台（如 Chrome 浏览器等）上无法显示 FVS 图形。尝试过了替换字形，插入空白字形等方式，都没能全平台上稳定地显示 FVS 图形。在现阶段基于 HarfBuzz 引擎，常规做法都无法绕过零宽限制。目前利用 HarfBuzz 的 Bug 机制，以非常规技术手段绕过零宽限制。实现了显示无效图形的功能。

十、关于第一音节的判断问题

1825（oe）/1826（ue）两个元音在第一音节当主体元音和非第一音节里当主体元音时字形是有区别。（注：双元音里的字形不受音节的影响）

蒙古文语法里常说的“在词里第一音节的主体元音”，其实就是编码序列里的“非词首第一元音”。在转换规则里判断是否为第一音节时候，建议使用：第一元音之前可以有无限个辅音的逻辑。

情况总结如下：

（1） 以下是第一音节，1825（oe）/1826（ue）应该以有长牙的字形显示。

计算：辅音{1,∞}+oe/ue+蒙古文字符

（2） 其他情况下认为非第一音节，1825（oe）/1826（ue）应该以没有长牙的字形显示。

十一、元音（A,I,O,U,OE,UE）双词根词词首字母字形是否阻断语法属性问题

在词中，当使用自由变体选择符请求出元音在多词根词第二及以后词根词首出现的字形时，表明之前的部分为一个单词的词末，这两个单词虽然连写，其实是两个独立单词。以上是常规语言学思考方式。将这个逻辑实现到转换规则当中后确实能够少用一个控制符。但是

(1). 在是否合体字、阴阳性传递等面存在不确定性，需要大量测试。

(2). 双词根词第二词根的首字母有时候不一定是元音，单独处理在这种情况下的双词根词语法属性阻断问题，可能会破坏规则的系统性和完整性。

所以目前暂未采纳这个逻辑。

十二、不建议在蒙古文中使用零宽度连接符 ZWJ 和定义 NIRUGU 尺寸属性

现在不同系统平台对 ZWJ 跟蒙古文混合使用时支持不够稳定。所以我们建议：

1. 在对 ZWJ 混合使用支持不够好的平台上建议使用 NIRUGU 来代替 ZWJ。
   ZWJ+<U+182D>+FVSn+ZWJ => NIRUGU+<U+182D>+FVSn+NIRUGU
   ZWJ+<U+182D>+FVSn+ZWJ => 
   NIRUGU+<U+182D>+FVSn+NIRUGU=> 
2. 如果在教学需求上需要明显展示词首、词中、词末孤立字形时，建议使用 NIRUGU，而不是 ZWJ。
3. 在孤立字母的两边使用 ZWJ 也有一定的不稳定性。而在单词中间使用 ZWJ 时候有更多的不稳定，很难达成各种平台上具有统一显示行为。所以不建议单词中间使用 ZWJ。字体厂家可以不考虑 ZWJ 出现在编码序列中出现的情况。

在蒙古文文本中使用 NIRUGU 的应用场景有延长基线、阻断自动组成合体字、创建断词（生成词里位置环境：上中下）等。依据这些需求，我们建议：

1. NIRUGU 长度设置为在词中约为一个长度，词首和词末约等于二分之一长度。一个长度单位为当前字体两个短牙根部之间的间距。但是字体厂可根据字体视觉需求的不同而定义个性化的长度。
2. NIRUGU 字形不是简单的一成不变的基线，而是根据词里位置，词里上下连接字形的不同而有所变形。比如手写体里，词首字形和词末字形就应该不同。字体厂家根据字体的美观性，可以改变长度。字体厂可根据字体视觉需求的不同而定义个性化的字形及字形变形机制。

已发现问题：

目前 NIRUGU 在 UCD 字符属性库中的属性跟普通蒙古文字母有所不同。所以在极少数系统软件中会出现不能自动变形的应用场景。如在 Linux 平台的 Libre 系列中，给紧连三个 NIRUGU 的中间一个设置不同颜色，那么会破坏连写环境，变成独立的字形。而同样操作不会破坏三个连起来普通蒙古文字母的连写环境。在 Win、Mac 系列引擎上没有发现这样的现象。这个可能只是文本编辑器层面的问题。所以目前忽略了这种差异，保持了常规变形机制。但是，为了今后稳定性，期望 NIRUGU 具有跟普通蒙古文字母相同的 UCD 字符属性。

十三、蒙古文空格尺寸要求及拉伸特征定义

在认同 MVS(U+180E)为一种蒙古文空格（仅仅视觉效果上）后，在蒙古文文本中可能出现的空格有三种：普通空格(U+0020)，NNBSP(U+202F)，MVS(U+180E，有三种字形)。对上述三个字符的字形制作建议为：

- U+0020：普通空格，SPACE。宽度等于普通空格，希望具有两端对齐排版时允许拉伸特征。
- U+202F：窄宽度无间断空格，NNBSP。宽度等于普通空格，希望具有两端对齐排版时允许拉伸特征。（本规则中不建议使用 NNBSP）
- U+180E：蒙古文元音分隔符（蒙古文空格、蒙古文附加成分分隔符），MVS。有两种常规使用需求：

  (1). 蒙古文元音分隔符：宽度约为普通空格的一半。承担现有 MVS 的

功能。希望具有两端对齐排版时禁止拉伸特征。

(2). 蒙古文附加成分分隔符：宽度为普通空格。承担现有 NNBSP 的功能。除蒙古文元音分隔符以外所有的场景都可以叫做蒙古文附加成分分隔符。希望具有两端对齐排版时允许拉伸特征。

※对同一个字母不同变体设置不同拉伸特征可能不符合要求。不过这里只是列出期望的属性，在基于现有编码方案，而通过最小最必要的改动，在最短时间内达到稳定可用状态的折中方法。

十四、将 MVS 和 NNBSP 的效用统一到 MVS

为了语言学上的［格附加成分不是独立词，是前边词不可分割的一部分］的语言学概念在编码层面实现，硬生生引入了 NNBSP，而它又不能好好地工作（不同平台不同系统对 NNBSP 参与蒙古文变形计算功能的支持情况一直不稳定）。如不同文本引擎 coretext,harfbuzz,directwrite 的显示差异、pdf 操作输出不统一、PowerPoint 上的编辑页面和预览页面都有不统一显示行为等。

其实用普通空格分割蒙古文格附加成分后，也可以叫做“在语言学上它是前边词的不可分割的一部分”，这样语言学概念和编码概念完全可以分开。但是既然已经维持了二十年的理论混搭，那么下边我们的主要工作还是维持这一“理论”的稳定实施而做的。即在编码层面彻底执行语言学层面的“不可分割的一部分”理论。

这次改变的另一个目的（便利性）是视觉操作体验的提升。任何控制符都是可见（有宽度空格也是可见的）的，操作体验非常方便。并且所有码位都在 1800 区域内。

解决方案：

在蒙古文应用场景中，附加成分的编码不再依赖 NNBSP（U+202F）,而由 MVS(U+180E)承担 NNBSP（U+202F）现有的全部功能（空格功能、变形控制功能）。即在编码层面直接替换。这是尽早摆脱 NNBSP 问题的最快方式。

期望的图形视觉效果：

在编码角度，MVS 是一个有宽度可见的普通蒙古文字母，依据上下文可变形，不具有控制字符属性。在用户使用角度，它有三种不同的字形：

(1). 蒙古文元音分隔符空格图形（有效图形）

在词中间有效地分隔了分尾元音 A/E，即蒙古文元音分隔符 MVS 的原有功能。条件为：在词末元音 A/E 之前出现。其尺寸定义参考【蒙古文文空格尺寸及拉伸特征定义】。

(2). 蒙古文附加成分分隔符空格图形（有效图形）

在词中间有效地分隔了认为“语言学上不可分离的一部分”的所有的一切。即为 NNBSP 的原有功能。条件为：除了词末元音 A/E 之前以外的场景。其尺寸定义参考【蒙古文文空格尺寸及拉伸特征定义】。

(3). 无效图形（错误图形）

错误使用时的指示图形。跟普通空格同样尺寸图形内部绘制线框图形。

※在 Unicode 码位备注里说明为 Mongolian Suffix Connector 蒙古文附加成分连接符（或分隔符）。这样既遵循了“不是独立词”原则，也提高了“视觉效果上的具有操作直观性等用户体验”。目的为 MVS 是可见可操作的。

存在不足：

(1). 不能满足格附加成分顶格需求

只要 MVS（其实 NNBSP 也存在同样问题）有宽度，那么就会附带引入“顶格空格”的副作用。比如制作牌匾时格附加成分无法顶格，因为顶部有一个空白空隙（就是 MVS 空格）。这种顶格需求是跟 MVS 可见可操作需求对立的，无法同时兼顾。除非制定“MVS 和 NNBSP 只提供不断词不断行功能，不参与变形计算，格附加成分的变形全依赖 FVS 选形功能”的原则。但是，这样又违背了“格附加成分不是独立词”的原则。或者定义“MVS 和 NNBSP 是零宽度不可见的”，但是这样也会跟“有宽度可见的便于视觉上的直接操作（光标走位等）”等的便利性发生对立。这就是一个需求多样性和技术实现的矛盾。

其实，“将不是独立词的格附加成分独立拆分出来顶格使用（行首和标点符号之后词首位置时零宽度不可见）”是一个不常规需求，既然不是独立词，那么独立顶格使用是一个特殊需求，既然特殊需求，那么应当特殊处理（如 FVS 选形、零宽度 MVS 风格字体、排版软件内部技术处理等）。所以我们现在建议采取方案为[期望效果]，并争取在国际标准层面修订完善相关属性修订工作。

其实这种零宽度顶格需求是可以通过制定“在词首（行首，其他文种之后）时显示为零宽度不可见字形”规则的方式来解决。但是这种为特殊需求而修改常规规则的方式，不建议采取。

(2). 编码层面分不清是格附加成分还是普通分尾（或其他普通单词）

在编码层面只提供技术实现机制，而不应该限制用户具体怎使用和语言学上个怎么解释。只要用户认为是附加成分的单词前边都可以加 MVS。这种语法（格附加成分，还是其它附加成分）层面的不同角色不应该记录在编码层面。现在是用 NNBSP 和 MVS 的不同来区分不同的语法角色，但是也没有完全区分开来。因为有些格附加成分用 NNBSP,而有些不用 NNBSP 而能够正确显示。我们不能要求必须全部使用 NNBSP。无论如何总会出现特殊，并且总会通过其它方式（字形判断，编码序列判断）来区分分尾、格附加成分、其它附加成分等等。总之将语法层面的角色用编码来区分是将问题复杂化，并且也没能解决好。

(3). 编码层面分不清是元音分隔符之后的分尾 A/E 和附加成分分尾 A/E

以前用 MVS 来判断分尾 A/E,那么现在也可以 MVS+A/E 组合来判断分尾 A/E。所以虽然是有一定的弊端，但是不至于是致命的，并且分尾 A/E 方式的附加成分不存在于现代蒙古文中。

※以前的分词判断逻辑是：先以空格分词获得独立词列表，然后以 NNBSP 分割获得主词和附加成分（包括格附加成分、类似 NUGUD 的附词、普通单词），最后再对 NNBSP 之后的编码序列进行详细对比后才确定格附加成分。如类似 NUGUD 的附词，还是普通蒙古文单词。原本 NNBSP 只是个窄宽度不间断空格，在文本中的任何地方都有可能用得到。所以即使在以前的方案上也无法回避【根据判断 NNBSP 之后编码序列来做语义区分】的工作。这次 MVS 和 NNBSP 合并后，也只是增加了一点点判断编码序列的逻辑，并没有额外增加一个新的环节，我们只需要多判断一个【MVS（等价于 NNBSP）之后编码序列是否为 A/E】。所以并没有实质性的增加难度，或破坏系统性。

(4). 两端对齐拉伸空格问题

期望效果是作为蒙古文元音分隔符（窄宽空格）时禁止拉伸，作为蒙古文附加成分分隔符（普通空格）时允许拉伸。但是这种基于字符串（或当前字形）的两端拉伸效果可能无法作为字符属性来定义。也就是无法做到所有软件的通

用性。但是现有 NNBSP 也同样没有做到全系统支持其两端对齐。

兼容性要求：

在[新字体+原来文本]上，在不改变文本编码的情况下，即支持原来[TEXT+MVS+A/E]和[TEXT+NNBSP+SUFFIX]方式的用法，也支持[TEXT+MVS+A/E]和[TEXT+MVS+SUFFIX]。并且排版布局不发生变化。支持 MVS 和 NNBSP 一对一替换。但是在输入法层面不提供 NNBSP 键位映射，尽量阻止使用。

期望效果的具体变形规则建议：

有如下三种字形：蒙古文元音分隔符空格图形，蒙古文附加成分分隔符空格图形，无效图形。

关于有效无效判断，有两种可行的方案。一、既然是 MVS 是有宽度可见（虽然是个空格图形）的，那么用户能看见图形的情况下，继续输入，那么是用户的有意之为。没必要在字体层面做有效无效的判断，允许用户连续使用。二，MVS 功能限定为蒙古文单词中只能使用一个，不能够连续使用。否则视为无效使用。目前现阶段建议采用后者：蒙古文单词当中只允许使用单个 MVS，连续使用的 MVS 都视为无效用法，即显示无效图形。后续可以升级字体方式完善连续使用功能。

现阶段具体建议如下（也是对转换规则的变形规则要求）：

(1). 无效图形

符合如下规则的场景，显示为无效图形。

- [词边界|FVS|ZWJ|MVS] + MVS + [词边界|FVS|ZWJ|MVS]

  这里需要排除例外：[蒙古文字母+N|H|G+FVSx+MVS+A/E +词边界]。

- [蒙古文字母]{1,∞}+ MVS + 词边界

  是一种无效用法，MVS 显示为【无效图形】。

(2). 蒙古文元音分隔符空格图形

符合如下规则的场景，显示为蒙古文元音分隔符空格图形。

- [蒙古文字母]{1,∞}|[词边界]+MVS+[蒙古文元音 A/E]+[词边界]

  即，只要在词末 A/E 之前的 MVS 永远以蒙古文元音分隔符（窄

宽度空格）形式显示。

(3). 蒙古文附加成分分隔符空格图形

除了上述两种场景以外的，都可以作为蒙古文附加成分分隔符空格图形来显示。为了提示建议，列举几个常规应用场景，但不一定是全部。

- [词边界] + MVS + [除 A/E 外的蒙古文字母]{1,1}
- [词边界] + MVS + [蒙古文字母]{2,∞}
- [词边界] + [蒙古文字母]{1,1} + MVS + [蒙古文字母]{1,∞}
- [蒙古文字母]{1,∞} + MVS + [蒙古文字母]{2,∞}
- [蒙古文字母]{1,∞} + MVS + [非 A/E 的蒙古文字母]{1,1}

十五、关于输入法键位设计的建议

此内容只是输入法键位设计的一种建议：在蒙古文中不建议使用 NNBSP，直接使用 MVS。建议英文减号（U+002D）键位上映射 MVS（U+180E），英文下划线（U+005F）键位（也是减号的 Shift 档键）上映射 NIRUGU(U+180A)。

十六、蒙古文字符 NIRUGU 及 FVS 的语法上下文传递行为

1. 关于蒙古文固定搭配字符序列

在蒙古文中有五种固定搭配字符序列，是不带任何自由变体选择符的蒙古文字母组合。这五种固定搭配字符序列中间使用 FVS 时其变形规则跟其它字符序列有所不同，所以在这里特殊说明。当蒙古文字符串的字符序列符合这五种固定搭配字符序列条件时，使用其特定的固定搭配字符序列默认变形规则，在转换规则内容中有详细描述。当在中间插入自有变体选择符时不能使用其固定搭配字符序列的默认变形规则（就是上述提到的固定搭配字符序列特有的变形规则）而使用常规语法上下文变形规则。

这五种固定搭配字符序列有：由 MVS(U+180E)前导的蒙古文附加成分；由[辅音+元音]形成的蒙古文十二字头；特定组合 1（[U+1826]+[U+1826]）；特定

组合 2（[U+1824]+[U+1824]）；特定组合 3（[U+182A]+[U+1826]+[U+1826]）。

2. 关于蒙古文字符 NIRUGU 语法上下文传递行为

蒙古文字符 NIRUGU 是个特殊字符，既不是元音，也不是辅音。它只创造位置上下文环境，但不创造语法上下文环境。无论哪种语法上下文环境（包括上述五种固定搭配字符序列的字符串中、包括能够组成合体字形的两个字符中间）下，蒙古文字符 NIRUGU 都具有无条件上下双向传递语法上下文的透明属性，也不会破坏任何已有特定变形规则。这种让 NIRUGU 在所有场景下都不破坏语法上下文的考虑点是：在蒙古文中 NIRUGU 的存在就是为了拉长单词基线，普通用户在单词中间插入 NIRUGU 的最大可能性的意图不是要改变什么字形，而是拉长基线。

※注：当圆头辅音 bpfkK 和词末元音 e 组合中间插入 NIRUGU 时透明处理，并且需要组成合体。但是词末元音 e 显示为不带短牙的左尾，不是带牙的左尾。

3. 关于蒙古文自有变体选择符语法上下文传递行为

蒙古文自有变体选择符（FVS1,FVS2,FVS3）不具有创造位置上下文环境的功能，它只能跟着当前字符使用。在使用上，也具有上下双向传递语法上下文的透明属性。但是在上述五种固定搭配字符序列中使用（无论词中间，还是单词最后追加）时，都会破坏这些固定搭配字符序列的专有默认变形规则，让其变成普通字符序列而使用常规语法上下文变形规则，这个跟 NIRUGU 的作用是有所不同的。这种让自由变体选择符在固定搭配字符序列中使用时需要破坏语法上下文的考虑点是：本来固定搭配字符序列在不用额外自由变体选择符的情况下，已经能够正确显示为符合正字法要求的情况下，用户还要插入自由变体选择符，那么普通用户的需求显然不是需要固定搭配字符序列的常规字形，所以应当让其使用普通字符序列的变形规则。当然，如果根据普通字符串的常规语法上下文变形规则来计算出的字形跟固定搭配字符序列的默认变形规则相同字形的话，是允许的。不是说既然破坏了固定搭配字符序列的默认变形规则，那么其通过常规字符串变形规则来计算的字形一定要跟其有差别。

在常规语法上下文变形规则中，当前字符按照自由变体选择符请求的字形形式显示，其余字符按普通字符串常规上下文变形规则要求来显示。在常规上下文变形规则中，自由变体选择符具有上下双向传递语法上下文的透明属性。

4. 转换规则描述内容的基于字母和基于变体的差异

在转换规则正文内容的正字法上下文描述当中，除了合体字及元音 i（U+1822）长牙有关以外的描述都是基于字母的，唯独这两个是基于变体（或变体显现字符）的。如规则[在词末，圆头辅音之后]不是基于字母的描述，而是基于变体的描述。这条描述的全部意义是[在词末，可以组成合体字形的圆头辅音变体之后]。如果通过后置自由变体选择符来强制请求出不可合体的圆头辅音是不符合这条规则的。即，转换规则内容当中的变形条件[圆头辅音之后]是指[这些圆头辅音在当前条件下，与后置元音能够组成合体字形]的语法上下文环境。不是所有[圆头辅音+FVS{0, ∞}+当前字符]都符合[圆头辅音之后]的变形条件。

5. 插入 NIRUGU 和无效 FVS 导致的虚拟合体问题

在单词中的 NIRUGU 和无效 FVS 是对包括合体字形等的语法上下文是透明的。但是，虽然不阻断组成合体字形，但是在视觉上隔离了两个变体字形。这导致[一个完整合体字形]比[虚拟连接的两个变体字形]是多出来一个单牙或长牙。这个多出来的单牙或长牙是属于前边辅音的，还是后边元音的没有明确定义。为了让虚拟链接的变体字形跟当前字符变体列表定义里的字形保持一致，我们规范成：当辅音+NIRUGU|无效 FVS+辅音组成虚拟合体字形时，元音以不带多出来短牙（或长牙）的原本位置变体形式显示。说明如下：

案例 1：

原生合体：ᠠᠪᠤᠪᠠ，　　1820+182A+1824+182A+1820

虚拟合体：ᠠᠪ᠍ᠤᠪ᠊ᠠ，　　1820+182A+FVS3+1824+182A+NIRUGU+1820

案例 2：

原生合体：ᠵᠠᠬᠢᠪᠢ，　　1835+1820+182C+1822+182A+1822

虚拟合体：ᠵᠠᠬ᠊ᠢᠪ᠍ᠢ，　　1835+1820+182C+NIRUGU+1822+182A+FVS3+1822

案例 3：

原生合体：ᠠᠪᠤ，　　1820+182A+1824

虚拟合体：ᠠᠪ᠍ᠤ，　　1820+182A+FVS3+1824

虚拟合体：ᠠᠪ᠊ᠤ，　　1820+182A+NIRUGU+1824

6. 十二字头固定搭配字符序列中插入自由变体选择符时变形原则

在处理传统蒙古文字符串中的自由变体选择符是否破坏固定搭配字符序列特定规则情况时，需要特别注意圆头辅音、非圆头辅音、十二字头（辅音+元音）、合体字形、字形阴阳属性等概念。需要考虑的各种场景（多余 FVS 等于有效 FVS 的一种，包含十二字头和常规词末音节）：

- 非圆头辅音+FVS+元音 A/E（阴阳性差异）
- 圆头辅音+FVS+元音 A/E（阴阳性差异）
- 非圆头辅音+FVS+元音 W/V
- 圆头辅音+FVS+元音 W/V
- 非圆头辅音+FVS+元音 O/U
- 圆头辅音+FVS+元音 O/U

在下边描述中用得到的专有术语定义如下：

蒙古文元音：

指编码在 U+1820、U+1821、U+1822、U+1823、U+1824、U+1825、U+1826、U+1827、U+1887、U+1888 范围内的一个蒙古文名义字符。

蒙古文辅音：

指编码在 U+1828-U+1842、U+1889-U+1897、U+1853、U+1858、U+185B-185C、U+18A6-18A7、U+18AA 范围内的一个蒙古文名义字符。

蒙古文圆头辅音：

蒙古文圆头辅音是指跟蒙古文元音字符连写时改变原来字形，与其融合成强制合体字字形的辅音字符。蒙古文圆头辅音包括 U+182A，U+182B，U+182C，U+182D，U+1839，U+183A，U+183B，U+1889，U+1892，U+1893，U+1858 等。

蒙古文非圆头辅音

除蒙古文圆头辅音字符以外的所有蒙古文辅音字符。

蒙古文十二字头

由[元音字母]或[辅音字母+元音字母]组成的字符串。

变形规则原则说明

上述场景都属于自有变体选择符破坏固定搭配字符序列特有变形规则的场景。对它进行变形计算时采用如下变形原则（对转换规则正文内容的补充定义）：

(1). 是否破坏固定搭配字符序列

判断字符串是否包含FVS,如果不是，那么使用固定搭配字符序列默认变形规则（基于[字母]）来显示，变形计算流程结束。如果是，破坏了固定搭配字符序列默认变形规则，进入下一步使用常规语法上下文变形规则的流程。

(2). 是否可以组成合体字形（先级级高）

判断当前字母变体与后置字母变体在字形上是否有组成合体字形的可能时，需要注意的是，这里说的是字母变体，不是字母，这有很大差异。固定搭配字符序列的默认变形规则是基于字母的，而是合体字形相关变形规则时基于字母变体（只考虑元音阴阳性）的。基于合体与否进入两种不同变形规则计算流程。关于是否合体，详细内容参考[合体字组合说明]文档。概要性解释参考如下：

- 圆头辅音bpfkK等的合体跟元音阴阳性无关。
- H/G辅音（无FVS，自动）+ 阴性元音 => 自动合体
- H/G辅音（无FVS，自动）+ 阳性元音 => 不合体
- H/G辅音（FVS强制请求阴性字形）+ 阴性元音 => 自动合体
- H/G辅音（FVS强制请求阴性字形）+ 阳性元音 => 不合体
- H/G辅音（FVS强制请求阳性字形）+ 阴性元音 => 不合体
- H/G辅音（FVS强制请求阳性字形）+ 阳性元音 => 不合体

(3). 创建变体组合清单

根据能否组成合体字形来，在上下两个字母的所有变体之间进行排列组合，获得最大范围的变体组合清单。有两种处理方式：

A. 能够组成合体字形时的变体组合清单

如果变体组合有组成合体字形的可能性，则排除后置字符的“不能组成合体字形”的变体，从“有可能组成合体字形”的变体中

基于“按字形组成合体”原则匹配（具体匹配规则参见具体字母的转换规则）出符合要求的变体组合清单，供下一步筛选。在这一步的运算结果只有一种类型：形成合体字形。

B. 不能组成合体字形时的变体组合清单

如果变体组合没有组成合体字形的可能性，则排除后置字符变体中的“为组成合体字形而定义”的变体，再从其它变体中匹配（具体匹配规则参见具体字母的转换规则）出符合要求的变体组合清单，供下一步筛选。在这一步的运算结果有两种：[非圆头辅音组成非合体字形的十二字头]字形和[其它]字形。

(4). 非合体十二字头相同字形回避原则

执行上述过程后，在变体组合清单里剩下如下三种类型的组合：

- 三个或以上字符组成的常规字符串

  不属于固定搭配字符序列范畴，按常规处理即可。

- 能够组成合体字形的两个字符组合

  由于合体字形是不受FVS影响，只考虑变体字形（计算FVS效用后）的合体可能性，优先级最高。所以针对这一类变体组合不做任何回避处理，按常规处理即可。

- 不能组成合体字形的两个字符组合（非合体十二字头字形）

  在这个可能性下，如果不加以差异化，最后[辅音+FVS+元音]字符串和[辅音+元音]字符串的字形是相同了，就无法实现为[破坏固定搭配字符序列的默认变形规则]。为了严格区分这两个字符串的字形差异，制定了[非合体十二字头相同字形回避原则]，既在不能组成合体字形的前提下，如果[辅音+元音]字形和[辅音+FVS+元音]字形相同，那么为了回避相同字形，[辅音+FVS+元音]主动采用与[辅音+元音]字形不相同的字形。

(5). 位置第一优先原则

如果剩下的变体组合数量还是大于一，或数量为零，那么使用位置变体列表中的第一个变体。

推理过程参考案例

案例 1：182A+FVS3+1820（  ）

(1). 由于 FVSx 的插入，破坏了原有 182A|182B|1839|183A|183B+1820 形式的固定搭配（十二字头）默认变形规则（辅音+元音），进而使用普通语法上下文变形规则（辅音+FVS+元音）。

(2). 由于针对字符 182A|182B|1839|183A|183B 来说上述 FVSx 是无效用法，需要显示无效 FVS 的示意图形。虽然不能跨过中间无效 FVS 图形跟后边元音组成合体字形，但是按照“无效 FVS 透明”原则，[辅音字形+元音字形]依然处于可以组成合体字形的上下文环境中。所以在能够组成合体字形的变体中选择。

(3). 1820/1821 共有三个词末字形。但是在上一步限定了能够组成合体字形的字母变体范围，所以排除了词末字形的第一（右尾）和第三（左分尾）。

(4). 这里不存在非合体十二字头相同字形回避问题。

(5). 最后只剩下词末第二字形（  ）可供选择。从最终结论来看，跟十二字头默认变形规则没有区别了，但是这是允许的。

※这里需要注意的是，常规合体时元音 1820 的左尾是带牙左尾，但是中间有了无效控制符（NIRUGU 也是相同推理）后，需要分开显示。但是元音

案例 2：182C+FVS1+1821（  ）

(1). 由于 FVS1 的插入，破坏了原有 182C+1821 形式的固定搭配（十二字头）默认变形规则（辅音+元音），进而使用普通语法上下文变形规则（辅音+FVS+元音）。

(2). 由于 FVS1 的请求，182C 显示为对应的阳性字形，不能与后面 1821 组成合体字形。所以词末 1821 排除专门为合体字形准备的词末第二字形后，在词末第一字形和第三字形之间选择。

(3). 但是很显然，词末第三字形的变形条件为[MVS+元音]，不符合这个规则的，所以被排除。

(4). 这里不存在非合体十二字头相同字形回避问题。

(5). 最后只剩下词末第一字形。从最终结论来看，跟十二字头默认变形规则没有区别了，但是这是允许的。

案例 3：182C+FVS2+1821（  ）

(1). 由于 FVS1 的插入，破坏了原有 182C+1821 形式的固定搭配（十二字头）默认变形规则（辅音+元音），进而使用普通语法上下文变形规则（辅音+FVS+元音）。

(2). 由于 FVS1 的请求，182C 显示为对应的阴性字形，可以与后续 1821 组成合体字形。所以词末 1821 的变体选形也被限定在能够组成合体字形的变体范围里了。

(3). 1821 共有三个词末字形，但是在上一步限定了能够组成合体字形的字母变体范围，所以排除了词末字形的第一和第三。

(4). 这里不存在非合体十二字头相同字形回避问题。

(5). 最后只剩下词末第二字形（ᡱ）可供选择。从最终结论来看，跟十二字头默认变形规则没有区别了，但是这是允许的。

案例 4：182D+FVS1+1826（᠊᠍᠖）

(1). 由于 FVS1 的插入，破坏了原有 182D+1826 固定搭配（十二字头）默认变形规则（非蒙古文字符+蒙古文辅音+1826），进而使用普通语法上下文变形规则（非蒙古文字符+蒙古文辅音+FVS+1826）。

(2). 由于 FVS1 的字形请求，182D 显示为无点阳性字形（ᡭ），不能跟后边元音组成合体字形。所以词末 1826 的变体选形被限定在不能够组成合体字形的变体范围里了。1826 共有三个词末字形。

※需要注意，虽然在 1826 词末第三字形的变形条件[圆头辅音之后]是指[能够形成合体字形的圆头辅音变体之后]。

(3). 其中，词末第三字形（ᡱ）是为[至少包含三个或以上字符的字符串]&[能够圆头辅音组成合体字形]的变形条件准备的。基于上一步判断，明显不符合此条，所以被排除。

(4). 其中，词末第二字形（ᡰ）是为[非圆头辅音组成非合体字形的十二字头]，或[圆头辅音组成合体字形]准备的字形。很明显，当前 182D 的[无点阳性字形]不满足[能够在正字法意义组成十二字头的非圆头辅音]，也不满足[能够组成合体字形的圆头辅音]，所以 1826 的词末第二（ᡰ）也被排除。

(5). 这里不存在非合体十二字头相同字形回避问题。

(6). 最后，只剩下词末第一字形（ᡯ）供选形了。

案例 5：182D+FVS2+1826（᠊᠎᠖）

(1). 由于 FVS2 的插入，破坏了 182D+1826 的十二字头的固定搭配默认变形规则。进而使用普通语法上下文变形规则。

(2). 由于 FVS2 的字形请求，182D 显示为阴性字形（ᡮ）。可以与后续元音组成合体字形，所以词末 1826 的变体选形被限定在能够组成合体字形的变体范围里了。1826 共有三个词末字形。

※需要注意，虽然在 1826 词末第三字形的变形条件[圆头辅音之后]是指[能够形成合体字形的圆头辅音变体之后]。

(3). 其中，很明显，词末第一字形（ᡯ）是不能组成合体字形的，所以被排除。

(4). 其中，词末第三字形（）是为[至少包含三个或以上字符的字符串]&[能够与圆头辅音组成合体字形]的变形条件准备的，明显是不符合，所以被排除。

(6). 其中，词末第二字形（）是为[非圆头辅音组成非合体字形的十二字头]和[与圆头辅音组成合体字形]准备的字形。很明显，当前 182D 的阴性字形是满足[能够组成合体字形的圆头辅音]，所以 1826 的词末第二（）是符合条件的。

(5). 由于行程了合体字形，所以不考虑[非合体十二字头相同字形回避原则]问题。

(6). 最后只剩下词末第二字形了。

案例 6：182D+FVS2+1823（）

(1). 由于 FVS2 的插入，破坏了原有 182D+1823 固定搭配（十二字头）默认变形规则（非蒙古文字符+蒙古文辅音+1823），进而使用普通语法上下文变形规则（非蒙古文字符+蒙古文辅音+FVS+1823）。

(2). 由于 FVS2 的字形请求，182D 显示为无点阴性字形（），虽然是阴性字形，但是辅音 182D 是不能跟阳性元音合体，所以词末 1823 的变体选形被限定在不能够组成合体字形的变体范围里了。1823 共有两个词末字形。

(3). 其中，词末第二字形（）是为[至少包含三个或以上字符的字符串]&[能够圆头辅音组成合体字形]的变形条件准备的，即[蒙古文字符串+圆头辅音+1823]。基于上一步判断，明显不符合此条，所以被排除。

(4). 其中，词末第一字形（）是为[非圆头辅音组成非合体字形的十二字头]，或[圆头辅音组成合体字形]准备的字形，即[蒙古文字符串+圆头辅音以外的辅音/蒙古文元音+1823]。很明显，当前 182D 的[无点阳性字形]满足[非圆头辅音]条件。依据[合体字组合说明]文档里的定义，这个字形是符合要求的。

(5). 最后只剩下词末第一字形（）供选形了。

案例 7：1831+FVS1+1826（）

(1). 由于 FVS1 的插入，破坏了 1831+1826 的固定搭配（十二字头）默认变形规则。进而使用普通语法上下文变形规则。

(2). 由于 FVS1 的字形请求，1831 显示为无点字形（），并且辅音 1831 不能跟后边的元音组成合体字形。所以词末 1826 的变体选形被限定在不能够组成合体字形的变体范围里了。1826 共有三个词末字形。

(3). 其中，词末第三字形（）是为[至少包含三个或以上字符的字符串]&[与圆头辅音组成合体字形]的变形条件准备的。基于上一步判断，明显不符合此条，所以被排除。

(4). 其中，词末第二字形（）是为[非圆头辅音组成非合体字形的十二字头]和[圆头辅音组成合体字形]准备的字形。很明显当前变体是符合[非圆头辅音组成非合体字形的十二字头]条件，需要保留此词末第二字形。但是，考虑到[非合体十二字头相同字形回避原则]，需要排除此字形。

(5). 最后只能选择词末第一（）字形了。

案例 8：1831+FVS3+1826（）

(1). 由于 FVS3 插入，破坏了 1831+1826 的固定搭配（十二字头）默认变形规则。进而使用普通语法上下文变形规则。

(2). 由于 FVS3 的字形请求是无效的，是一种错误用法。1831 依然显示为有点字形（），并且辅音 1831 不能跟后边的元音组成合体字形。所以直接排除了 1826 的词末第三字形（），只能在第一（）和第二（）词末字形中选形。

(3). 其中，词末第三字形（）是为[至少包含三个或以上字符的字符串]&[与圆头辅音组成合体字形]的变形条件准备的。基于上一步判断，明显不符合此条，所以被排除。

(4). 其中，词末第二字形（）是为[非圆头辅音组成非合体字形的十二字头]和[圆头辅音组成合体字形]准备的字形。很明显当前变体是符合[非圆头辅音组成非合体字形的十二字头]条件，需要保留此词末第二字形。但是，考虑到[非合体十二字头相同字形回避原则]，需要排除此字形。

(5). 最后只能选择词末第一（）字形了。

案例 9：1822+1837+1821+182C+FVS3+1826（）

(1). 不是固定搭配字符序列，所以不符合上述五种固定搭配字符序列的默认变形规则。使用普通语法上下文变形规则。

(2). 对于字母 182C，由于 FVS3 的字形请求，182C 直接显示为阳性（）字形，不能跟后边元音 1826 组成合体字形。

(3). 对于字母 1826，由于前边 182C 已经变为阳性字形，1826 不能跟其组成合体字形，所以词末 1826 已经不满足[圆头辅音之后]的变形条件。因此 1826 词末字形排除了第二（）和第三（）字形。

(4). 最后，词末 1826 只能显示为词末第一（）字形。

案例 10：1822+1837+1821+1831+FVS1+1826（）

(1). 不是固定搭配字符序列，所以不符合上述五种固定搭配字符序列的默认变形规则。使用普通语法上下文变形规则。

(2). 对于字母 1831，由于 FVS1 的字形请求，1831 直接显示为无点（）字形，并且辅音 1831 没有合体可能性。

(3). 对于字母 1826，由于 1826 不能跟上边字母组成合体字形，所以词末 1826 已经不满足[圆头辅音之后]的变形条件。因此词末 1826 只能在词末第一（）和第二（）字形之间选形。

(4). 由于 1826 词末第二字形是专门针对[辅音+元音]两个字母组合的十二字头准备的变形规则，当前由两个以上字母组成的字符串，显然不满足其变形条件，所以被排除。

(5). 最后，词末 1826 只能显示为词末第一（）字形。

案例 11：1836+1821+182C+FVS2+1826 （  ）

(1). 不是固定搭配字符序列，所以不符合上述五种固定搭配字符序列的默认变形规则。使用普通语法上下文变形规则。

(2). 对于字母 182C，由于 FVS2 的字形请求，182C 直接显示为词中阴性字形。

(3). 对于字母 1826，由于 1826 可以跟上边字母组成合体字形，所以词末 1826 已经满足[圆头辅音之后]的变形条件。因此词末 1826 在词末第二（）和第三（  ）字形之间选形。

(4). 由于 1826 词末第二字形是专门针对[辅音+元音]两个字母组合的十二字头准备的变形规则，当前由两个以上字母组成的字符串，显然不满足其变形条件，所以被排除。

(5). 最后，词末 1826 只能显示为词末第三（）字形，并且合体。

7. 关于透明属性的规则描述表达式

※这里只包含被认为是具有透明属性时 NIRUGU 和 FVS 的透明算法。不包含附加成分、十二字头、三个特定组合等三种固定字符序列的变形逻辑。FVS 代表 FVS1、FVS2、FVS3 的任意，FVS1、FVS2、FVS3 只代表当前一个自由变体选择符。

(1). 1822 词中形

[a|e|o|u|E]+[NIRUGU|FVS]{0, ∞}+i+蒙古文 → i 为词中第三形（双长牙）

(2). 1822 词中形

[i]+[NIRUGU|FVS]{0, ∞}+i+蒙古文 → i 为词中第一形（一个长牙）

(3). 1822 词中形

[辅音]+[辅音|NIRUGU|FVS]{0, ∞}+[oe|ue]+[NIRUGU|FVS1|FVS2]{0,∞}+i+蒙古文 → i 为词中第一形（一个长牙）

(4). 1822 词中形

[辅音]+[辅音|NIRUGU|FVS]{0,∞}+[oe|ue]+[FVS3]+[NIRUGU]{0,∞}+i+蒙古文 → i 为词中第三形（双长牙）

(5). 1822 词中形

[辅音]{0, ∞}+[元音]+[辅音|元音|NIRUGU]{0, ∞}+[辅音]+[NIRUGU]{0,∞}+[oe|ue]+[NIRUGU|FVS3]{0,∞}+i+蒙古文 → i 为词中第三形（双长牙）

(6). 1822 词中形

[辅音]{0, ∞}+[元音]+[辅音|元音|NIRUGU]{0, ∞}+[辅音]+[NIRUGU]{0,∞}+[oe|ue]+[FVS1|FVS2]+[NIRUGU]{0,∞}+i+蒙古文 → i 为词中第一形（一个长牙）

(7). 1823,1824 十二字头词末

词首辅音+[NIRUGU]{0, ∞}+[词末 o|词末 u] → o,u 为词末第二形

(8). 1823,1824 十二字头词末

词首辅音+[FVS]{1,∞}+[NIRUGU]{0, ∞}+[词末 o|词末 u] → o,u 为词末第一形

(9). 1825,1826 十二字头词末

词首辅音+[NIRUGU]{0, ∞}+[词末 oe|词末 ue] → oe,ue 为词末第二形

(10). 1825,1826 十二字头词末

词首辅音+[FVS]{1,∞}+[NIRUGU]{0, ∞}+[词末 oe|词末 ue] → oe,ue 为词末第一形

(11). 1825,1826 词中形，第一音节判断

词首辅音+[辅音|NIRUGU|FVS]{0,∞}+[oe|ue]+[FVS1]{0,1}+[蒙文字符|NIRUGU] → oe,ue 为词中第二形（带长牙）

(12). 1825,1826 词中形，第二音节及以后音节判断

[词首字符]+[词中元音]{1, ∞}+[辅音|NIRUGU|FVS]{0,∞}+[oe|ue]+[FVS3]{0,1} +[蒙文字符|NIRUGU] → oe,ue 为词中第一形（不带长牙）

(13). 1828 词中形

[词首字符|词中字符|NIRUGU|FVS]+n+[NIRUGU|FVS1|FVS3]{0,∞}+[词中元音|词末元音] → oe,ue 为词中第二形（带点）

(14). 1828 词中形

[词首字符|词中字符|NIRUGU|FVS]+n+[NIRUGU|FVS2|FVS3]{0,∞}+[词中辅音|词末辅音] → oe,ue 为词中第一形（不带点）

(15). 182D 词中形

[词首字符|词中字符|NIRUGU|FVS]+sa|da+[NIRUGU|FVS]{0,∞}+ga+[NIRUGU]{0,∞}+[词中 a|o|u 元音|词末 a|o|u 元音] → ga 为词中第一形（不带点）

(16). 182D 词中形

[词首字符|词中字符|NIRUGU|FVS]+sa|da+[NIRUGU|FVS]{0,∞}+ga+[MVS]+[词末 a 元音] → ga 为词中第六形（不带点）

(17). 1831 词首形

[词首 sha]+[NIRUGU|FVS2|FVS3]{0,∞}+i+蒙古文字符 → sha 为词首第二形（不带点）

(18). 1831 词首形

[词首 sha]+[NIRUGU|FVS1|FVS3]{0,∞}+a/e/o/u/oe/ue/E+蒙古文字符→ sha 为词首第一形（带点）

(19). 1831 词中形

[词首字符|词中字符|NIRUGU|FVS]+sha+[NIRUGU|FVS2|FVS3]{0,∞}+[词中 i|词末 i] → sha 为词中第二形（不带点）

(20). 1831 词中形

[词首字符|词中字符|NIRUGU|FVS]+sha+[NIRUGU|FVS1|FVS3]{0,∞}+[词中 a/e/o/u/oe/ue/E|词末 a/e/o/u/oe/ue/E] → sha 为词中第一形（带点）

(21). 1833 词中形

[词首字符|词中字符|NIRUGU|FVS]+da+[NIRUGU|FVS1|FVS3]{0,∞}+[词中元音|词末元音] → da 为词中第二形（平头）

(22). 1833 词中形

[词首字符|词中字符|NIRUGU|FVS]+da+[NIRUGU|FVS2|FVS3]{0,∞}+[词中辅音|词末辅音] → da 为词中第一形（非平头）

8. 关于蒙古文自由变体选择符透明属性的最佳方案

对于字符串中间的自有变体选择符，无论有效还是无效，应当采用如下方案：

1. 规则一定要区分是基于字母的，还是基于变体的。应当每条规则下边都要标注。
2. 变形规则是基于字母的，FVS 永远只影响当前字母，优先级最高。只有在处理合体字形等基于变体的变形规则时，FVS 才会变相影响其他字母的变形，其它时候，对于当前字母以外的其它字母来说 FVS 用永远是透明的。例如，字符串[字母 1+字母 2+字母 3]和[字母 1+FVS+字母 2+字母 3]中，对于字母 2 和字母 3 来说，都是使用相同的固定搭配变形规则，只要字母 1 使用强制请求字形。

这次未采纳。这是一个最大的遗憾！

（结束）