Meta
Journal des traducteurs
Translators' Journal

META

Inuktitut Syllabics and Microcomputers

Doug Hitch

Volume 38, numéro 1, mars 1993

La traduction et l'interprétation dans le nord du Canada
Translation and Interpretation in Northen Canada

URI : https://id.erudit.org/iderudit/003072ar
DOI : https://doi.org/10.7202/003072ar

Aller au sommaire du numéro

Éditeur(s)

Les Presses de l'Université de Montréal

ISSN

0026-0452 (imprimé)
1492-1421 (numérique)

Découvrir la revue

Citer cet article

Hitch, D. (1993). Inuktitut Syllabics and Microcomputers. *Meta*, *38*(1), 56–72. https://doi.org/10.7202/003072ar

Résumé de l'article

Le traitement de texte avec orthographe syllabique est maintenant très répandu. En 1985, un code informatique standard comme le ASCII fut proposé pour le syllabique afin de faciliter les communications. Il ne fut pas très largement accepté et il ne semble pas promis à beaucoup d'avenir. Les ordinateurs Macintosh ont toujours eu la capacité intrinsèque de reproduire le syllabique sur leur écran. Les ordinateurs de type DOS ont utilisé diverses technologies pour y parvenir. Pour les deux types d'appareils, il existe des traitements de texte en syllabique utilisant le standard proposé et d'autres qui ne l'utilisent pas. De nos jours, le Macintosh est l'outil privilégié pour travailler en syllabique. Trois différentes approches sont actuellement utilisées sur le Macintosh, soit un traducteur de clavier, un système de double frappe ou les touches "option". Le marché offre pour le Mac un choix de quatre fontes possibles. Deux organismes, ISO et Único Inc., travaillent présentement à la mise au point d'un nouveau code informatique qui contiendrait plus de 65 000 caractères. Le syllabique devrait être inclus dans cette trousse. Il serait utile de standardiser le clavier pour le syllabique. Il existe, pratiquement, une disposition des touches pour chacune des alternatives disponibles. Une disposition standardisée de l'orthographe syllabique, comme pour l'anglais, survivrait sûrement à plusieurs générations de changements technologiques.

INUKTITUT SYLLABICS AND MICROCOMPUTERS[1,2]

DOUG HITCH
Inuktitut Linguist, Department Responsible for Education, Culture and Employment Programmes, Government of the NWT,[3] Yellowknife

Résumé

Le traitement de texte avec orthographe syllabique est maintenant très répandu. En 1985, un code informatique standard comme le ASCII fut proposé pour le syllabique afin de faciliter les communications. Il ne fut pas très largement accepté et il ne semble pas promis à beaucoup d'avenir.

Les ordinateurs Macintosh ont toujours eu la capacité intrinsèque de reproduire le syllabique sur leur écran. Les ordinateurs de type DOS ont utilisé diverses technologies pour y parvenir. Pour les deux types d'appareils, il existe des traitements de texte en syllabique utilisant le standard proposé et d'autres qui ne l'utilisent pas.

De nos jours, le Macintosh est l'outil privilégié pour travailler en syllabique. Trois différentes approches sont actuellement utilisées sur le Macintosh, soit un traducteur de clavier, un système de double frappe ou les touches «option». Le marché offre pour le Mac un choix de quatre fontes possibles.

Deux organismes, ISO et Unico Inc., travaillent présentement à la mise au point d'un nouveau code informatique qui contiendrait plus de 65 000 caractères. Le syllabique devrait être inclus dans cette trousse.

Il serait utile de standardiser le clavier pour le syllabique. Il existe, pratiquement, une disposition des touches pour chacune des alternatives disponibles. Une disposition standardisée de l'orthographe syllabique, comme pour l'anglais, survivrait sûrement à plusieurs générations de changements technologiques.

Abstract

Word processing with Syllabics is now very common. Many different approaches have been used. In 1985 a computer code standard like the ASCII was proposed for Syllabics in order to facilitate communication. This has not been widely implemented and is not likely to gain further recognition.

Macintosh computers have always had a built-in ability to show Syllabics on the screen. DOS computers have employed various technologies to do this. For both types of computers there are Syllabics word-processing solutions that employ the proposed standard and those that do not.

Today the Macintosh is the machine of choice for work with Syllabics. Three different strategies are currently in use with the Macintosh, involving a keyboard translator, overstriking, or Option key. There are four outline fonts for the Mac on the market.

Two organizations, the ISO and Unicode, inc., are working on a new computer code which will contain more than 65,000 characters. Syllabics should be included in this set.

It would be useful to standardize the Syllabics keyboard. There are different key layouts for almost every solution. A standard layout for Syllabics, like that for English, will probably survive through several generations of technological change.

INTRODUCTION

The computer is now an integral part of the world in which we live. It is not only vital for business, education and government but can also play a key role in the promo-

tion of aboriginal languages. Microcomputers were being used to produce documents in Inuktitut syllabics almost as soon as they were available. Now very little work is done on typewriters.

This article examines the past, present and future of microcomputing with syllabics. Off-the-shelf computers and mass market software do not currently have the ability to use syllabics. Special software must be designed to provide it. Different solutions have come and gone and there are currently several competing ones.

COMPUTER CODES

Most of us do not need to think about how a computer works. We press the key for the letter *a*, an *a* will appear on the screen and an *a* will print on paper. Actually, for this simple thing to work, three different programs need to be coordinated. The scheme involves assigning a code number to a character and having the three programs agree on that code. The current code number[4] for *a* is 97.

Pressing a key sends a scan code from the keyboard to the Central Processing Unit or CPU (± 'electronic brain'). The CPU uses a keyboard map, the first piece of software, to translate the scan codes into computer codes. When you press the *a* key, the CPU and keyboard map translate this to decimal 97. The 97 is sent to the second program, a screen driver, which causes pixels in the screen to form the shape of an *a*. When you print a document, a string of codes is sent to the printer which it interprets and then places dots on the paper to form characters. When it receives the code 97 it prints an *a*.

For persons working in English, there is no need to be concerned about computer codes since the industry aims at the English market and the industry makes sure that the needs of the market are met. From the beginning there has been consensus to give *a* the code 97. When you buy a computer you know that you just type an *a* to get an *a*. With aboriginal languages the situation is different. The computer industry has essentially played no role in deciding the codes to be given to special characters such as Inuktitut syllabics. Anyone who develops software for syllabics is currently free to assign any code to any syllable. Thus today the final *v* sign, ^{e}, is assigned four different codes by four different Macintosh syllabics fonts. It is 61 in *Naamajut*, 43 in *Iqaluit+*, 123 in *Emilia*, and 223 in *Inuktitut (MacTitut* version). Computers using English can exchange information easily since an *a* is always 97. Computers using syllabics cannot exchange information without special effort to translate the codes. From the beginning of microcomputing with syllabics the code issue has been important.

ASCII AND THE CURRENT 256 CODE SET

Today's microcomputer uses a standard set of codes called *ASCII* which stands for the American Standard Code for Information Interchange. Developed in the 1960's with U.S. government encouragement, ASCII was more or less based on the character set used for the American typewriter and intended as a replacement for the Teletype code for sending information electronically. It was not specifically intended for use in computers nor did it take much account of the needs of languages other than English for transmitting information.

The original ASCII had 128 codes and all microcomputers follow this standard. However, for some time microcomputers have worked with a second set of 128, making a total of 266. The second set is not standardized and can contain any characters which the equipment manufacturer deems appropriate. This is where you can find the French letters *é* and *à*, the German *ü* and *ö*, Greek π and μ, or symbols like ¢ and †. The sets of three companies, Macintosh, IBM and Epson (a leading printer maker) have gained prominence and most manufacturers will make their products compatible with one of these sets.

The 256 code set currently used by computers is based on 8-bit communications technology which is now relatively outdated. All computers communicate in packets of *bits,* confusingly called *bytes.* For 8-bit computers the bytes naturally contain 8 bits. There are 2^8 or 256 distinct bytes. Each byte is equivalent to a code. Currently, most microcomputers use 16- or 32-bit communications. A 16-bit machine can use 216 codes, or, in other words, can recognize 65,536 separate codes. A single step in technology has astronomically increased the potential for character processing from 256 to 65,536, or by more than 500 times.

FUTURE CODE SETS

Two organizations are currently at work to develop a standard code set to take advantage of the new technology. The idea is to try and incorporate as many signs from as many different writing systems as possible. The set will certainly incorporate all of the characters used to write Chinese, Japanese, Russian, Greek, Arabic, Hebrew, Ethiopian, Hindi, Cambodian and many, many other languages, and will even include the hieroglyphics used for ancient Egyptian.

The International Standards Organization in Geneva was the first group to begin work on the 65K (65,536) code set. Because representatives of many countries are involved and because the body is probably trying to be as fair as possible, the process of collecting characters is slow. The computer industry itself has responded by setting up its own project to establish a character set called *Unicode.* All of the major companies including IBM, Apple, Microsoft, Xerox and Sun are taking part. The consortium is known as Unicode Inc. and operates out of Metaphor Computer Systems of Mountain View, California. It has apparently already assigned 27,000 codes. The first version has been released but I have been unable to track down substantive information on it.

In 1992 the Unicode and ISO 10646 standards stopped competing. The ISO recognized that Unicode would become an industry standard and in the meantime it had decided to develop a 32-bit standard instead with the potential for cataloging 4,294,967,296 characters. The ISO will now incorporate Unicode at the beginning of its 10646. Macintosh announced in 1992 that System 7.1 would work with Unicode. Its software is to be called *WorldScript.* This was to be available by October. I obtained a copy of 7.1 and was disappointed that WorldScript was absent. The Apple Canada 7.1 support line said that 7.1 is WorldScript aware but they did not know anything further.

It would be very useful if Inuktitut syllabics were in the new code set. This would mean that no matter where you would be in the world you could sit down behind any computer and work in syllabics with no extra software. The technical advantages will aid the promotion and development of the language. There are also political advantages that come with the greater visibility of the language. The world would recognize syllabics as a legitimate writing system and would know that people are actively using it.

CANADIAN ABORIGINAL SYLLABICS ENCODING COMMITTEE

On November 26 and 27, 1990 an information sharing session entitled *Native Language Syllabics and Computer Encoding* was held in Winnipeg. I was able to attend. The discussion revolved around the merit of standardizing computer codes for syllabics. There were two main points. First was the question of implementing a standard today, using the available 256 codes. This point will be discussed in detail shortly. The second issue involved the future 65K ISO standard. There was overwhelming enthusiasm for the idea of including syllabics in this truly international computer code.

The meeting was facilitated by Louise Campbell of the federal Department of Communications. Dirk Vermeulen of the Kativik School Board made important contributions to the agenda. Sometime after the meeting a group called the Canadian Aboriginal Syllabics Encoding Committee (CASEC) was set up with Dirk Vermeulen as chair and this has met fairly regularly. One of the tasks has been to prepare a 'repertoire' or 'repertoires' of Canadian aboriginal syllabics for submission to the Canadian Standards Association, which will then present the results to the International Standards Organization in Geneva and to Unicode. Dirk told me last October (1992) that a Unicode representative was at one of the recent Winnipeg meetings. The will appears to be there on the Unicode side. If a bottleneck remains in this process it is in Canada.

As a kind of associate member of CASEC I receive periodic updates on progress. Dirk Vermeulen represents CASEC at a technical committee of the CSA working with character sets. From the November 19, 1992 minutes of a meeting of that committee CASEC reported that three preliminary repertoires of syllabics had been created, one each for Inuktitut, Algonquian and Athabaskan. Work on naming the characters was continuing. Eventually these will be combined and submitted for development of a Canadian National standard and also input to ISO 10646 and Unicode.[5]

From the beginning the Department of Communications has sought to turn the initiative in the move towards standardization over to aboriginal organizations. The project management team now consists of a Project Manager who is Esther Wesley[6] of the National Association of Cultural Education Centres, a Technical Advisor who is Paul Green[7] of NACEC and the First Nations Technical Institute, and a Communications Canada representative, Andy Kwan.[8]

THE QUEST FOR A 256 CODE STANDARD

It is not clear how long we have before the 65K code set gets widely implemented. In the meantime we are still using a set of 256 codes and it is probably worthwhile to try and improve our use of it in the interim.

More than six years ago it was generally recognized that work on syllabics with computers would be greatly facilitated if a standard coding arrangement could be adopted. In March of 1985 a Syllabic Computer Workshop was held in Montreal by the Kativik School Board.[9]

> The meeting was sponsored by the department of Indian and Northern Affairs in order to respond to concerns that Syllabic software was being developed independently in different areas of the country with little communication between software developers. There was concern that this could lead to overlap in software development and that it also could lead to problems in compatibility between software packages and problems in communications between various types of computers.[10]

These fears have all been realized. There is overlap in software development and there is no widely used set of syllabics computer codes that could be considered a standard.

THE SUPERBRAIN CODES

Bob Johnson of Kirk Computer Systems Ltd., Yellowknife, presented to the workshop the ASCII code assignments used for the Intertech Superbrain.

> After long discussion of the potential benefits and dangers to the proposed standardization, the group agreed to resolve that the standard proposed be adopted as an interim working convention as accepted by the computer workshop.[11]

The coding arrangement proposed by Bob Johnson in 1985[12] would limit the syllabic assignments to the upper 128 codes. The reasons for this are discussed below. Since the smallest syllabary, the ICI Inuktitut standard, requires 105 codes,[13] this leaves at most 23 codes available for all the other variants. It might also be desirable to make provision for the */ai/* series in Inuktitut, the fourth column which is linguistically unnecessary but part of the culture and is still found in older materials as well as in religious literature and is still used by Elders. This might require 15 more codes.[14] At this point we have at most 8 codes left for other possible additions to the Inuktitut syllabary AND to satisfy all the Cree, Ojibway, OjiCree and Dene variants. Clearly, 128 codes is not enough for a "standard" syllabary that would satisfy everyone. Presumably it was at least partly for this reason that Bob Johnson proposed in 1989 that there be three standard Syllabaries, one for Inuktitut, one for Eastern Cree and one for Western Cree.[15] Even these have linguistic and technical problems as pointed out by Johnson.[16]

SUPERBRAIN AND MAC

When Bob Johnson proposed that the Superbrain convention be adopted, he was perhaps not taking fully into account the advanced font capabilities which the graphic interface of the Macintosh provides. This interface allows one to see different font types and sizes in one text and even to combine text and pictures easily in word processing documents. Only recently has the DOS world been able to offer a similar thing with the release of Windows 3.0 and now 3.1. Bob's proposal meant assigning all the syllabics graphs to the upper character set leaving the lower ASCII range with the usual English values. At that time not all software recognized the upper set which meant that extra programming costs would have to be incurred for implementation of the standard. He wrote:

> These problems mean that if the proposed standard is accepted, programming which uses Syllabics characters will be more costly and difficult than programs which use Roman only. I believe that such costs are made necessary by the basic requirement that a text file contain both Syllabics and Roman characters intermixed. Such texts are common today, not only in bilingual texts, but also when a word or name is "borrowed" into a Syllabics text, and left in Roman because it is difficult or impossible to translate.[17]

In essence, a main motivation for putting all the syllabics characters into the upper 128 codes was so that a document could contain both Roman and syllabics characters. Most microcomputers, including the Superbrain and DOS machines, could work with only one character set and that set had to be part of the hardware, encoded onto a (ROM) chip. A little later, by using a graphics card and special software the need for a customized (ROM) chip was overcome, but the number of different characters visible on the screen was still limited to 224[18] and one could not see different font sizes.

In contrast, the Macintosh with its graphic interface had no such limitations. Already in 1985 one could have Arabic, Ethiopian, Greek, Inuktitut, and Russian letters on the screen and on paper. The machine simply treated all of these as 'fonts' like *Geneva* or *Helvetica* or *Zapf Dingbats*. Font development packages were available from Apple, free of charge to non-profit organizations, and very quickly there appeared fonts on the Macintosh in every imaginable script including syllabics.

It is ironic that the one machine which had no trouble producing texts in both syllabics and Roman, the Macintosh, is also the one machine which more than any other has produced syllabics texts following the Superbrain convention. An important original reason for restricting the syllabics to the upper codes did not apply to the Mac and yet it is Mac users, specifically those working through *MacTitut* (see below), who, more than any others, have accepted the responsibility of using the convention.

I have shown Bob Johnson earlier versions of this paper and have had several discussions with him. He has brought up other useful points which are worth considering. One is that restricting syllabics to the upper codes means that the distinction between English and syllabics would be maintained in a text file. A text file contains no formatting of any kind and is used to send files between different types of computers, between different types of software on the same computer, and in telecommunications. Since there is no character formatting in a text file, any original font distinction is lost when a file is converted to text format.

In the April of 1991 version of this paper I argued that text files are getting less important as time moves on. It is getting more common for average users to transmit fully formatted files or files of any kind over networks and telephone lines. The point was that it was getting easier and safer to use the lower 128 codes for non-ASCII characters. This discussion will be irrelevant when Unicode is implemented.

Also at that time I was unaware of the ease at which key remapping could be done on the Mac (and now under Windows 3.1). I thought it acceptable to sacrifice the inviolability of the lower ASCII for the sake of a simple, economical way of doing syllabics. However, since discovering the joys of key remapping, and learning more about other syllabics computing tasks, I have adopted essentially such a character set for use until Unicode comes out (see below: Inuit ASCII). This set keeps the lower 128 intact, with two exceptions, and so follows the spirit of Superbrain.

Bob had another point which should be discussed. This has to do with the alphabetical ordering of syllabics. I did not understand the relevance of this when I first read the *Report on the Syllabic Computer Workshop*[19] and *A Proposal for Computer Encoding of Syllabics Text*.[20] When computers sort English text, they sort by ASCII code, from lowest to highest. The words come out in alphabetic order because the sequence of ASCII codes matches the sequence of the alphabet. Early microcomputers also sorted the upper 128 codes in the same fashion, from lowest to highest. Bob intended to order the sequence of syllabics codes to match the alphabetic ordering of the syllabary. At the time of his proposal this would have made sorting automatically work properly for syllabics. However, much of the more recent software now does not sort the upper 128 codes sequentially, rather, it will arrange some of the higher codes among the lower ones in order to follow the alphabets of other languages. For instance, it is getting more common to find software that sorts French *é* (142) right after *e* (101), German *ü* (159) right after *u* (117), etc., following the sequence of the French and German alphabets. Besides this, different software on the same machine will have different sort orders. Microsoft File, Word 4.0 and 5.0 all sort differently. The structure of the code set is no longer really relevant to sorting. Whatever the structure, special sorting software needs to be developed.[21]

A final point brought up by Bob is that it is not necessary that a machine use the same codes to work internally as it uses to communicate with other machines. It is technically possible for every user to have his own set of codes for their own work, and then to translate these into a standard set when communicating with another user who could then translate from the communications standard into whatever codes they happen to be using. But while it is possible in this way to have a standard it would be diffficult, timeconsuming and expensive to implement. Every user would need the ability to translate to and from the standard code set.

In spite of its adoption at the workshop in 1985, the interim working convention has not received general recognition. Nevertheless, various attempts have been made to use the codes. For both the Macintosh and DOS machines (IBM clones) there have been syllabics strategies that involve the interim working convention as well as others that do not. It is worthwhile here examining all of the solutions of which I am aware.

DOS COMPUTERS AND SYLLABICS

In the DOS world, not only have different programmers produced different syllabics software, but successive evolutionary developments in the DOS computer technology itself have brought changes in programming strategy, especially that used to get syllabics on the screen. Three technological stages in DOS screen programming can be identified. The earliest solution required that a special (ROM) chip be made and installed in the computer in order to generate syllabics on the screen. The only manufacturer appears to have been BECE Electronics in Grimsby, Ontario.[22] This company followed the interim working convention. In 1991 I knew of only two places where these chips were still in use. The Kativik School Board had about 100 such machines in operation (besides 25 Macintoshes), and a translation contractor in Yellowknife had 2. This week I learned that the Housing Association in Iqaluit also had a BECE equipped machine at one time but probably now uses a Mac for syllabics correspondence. It is unlikely that there are in current use many other computers with BECE character generators. There may be none.

The second stage in DOS screen technology was made possible by the use of graphics cards. Through software one could change the screen appearance of the basic character set. That is, one could replace the basic English set with any other. It is not necessary to use a ROM chip. However, to the best of my knowledge, only one set at a time could be shown on the screen, and different point sizes could not be shown, nor could bold or italic faces or underlining. I am not familiar with any second stage solution that uses the interim working convention. If they do exist they are probably not commercially distributed. There is however a commercial product employing this technology but it does not use the code set of the interim working convention. The Multi-Lingual Scholar, uses the lower codes for at least some syllabics graphs while the convention restricts all syllabics to the upper 128. Several dozen copies were purchased in the Keewatin region and as of 1991 were still used in Adult Learning Centres to teach word processing. I spoke to Mike Shouldice of Arctic College, Rankin Inlet, about this in the fall of 1990. At that time he thought that the syllabics version of Multi-Lingual Scholar first appeared two or three years earlier. It is still being sold and an occasional enquiry about it still crosses my desk.[23]

The current strategy for getting syllabics on DOS screens is based on the graphic interface now provided by Windows 3.0 or 3.1 or a similar software. This provides DOS users with much the same screen capabilities as those long enjoyed by Macintosh users. Now it is possible to see several different fonts, different point sizes, bold, italics and underlining on the screen of a DOS computer. Bob Johnson is working with syllabics and Windows and as far possible he is using the convention.[24] His product has the ability to change key mapping on the fly. It is very Mac-like. He has not made this solution commercially available. It uses the same typing method as MacTitut (see below: History of MacTitut).

Last week (Feb. 12, 1993) I received a fax from Susan Sammons containing a pamphlet advertising a syllabics for Windows product from Datarctic of Iqaluit. I called the developer, Peter Baril for more information. DIS©93, Datarctic Inuktitut Syllabi©s, is simply three Windows outline fonts. According to Peter it is a 'quick and dirty' product. It uses the most simple and direct way of producing high-quality syllabics on a laser printer. He restricted the syllabics characters to the lower 128. Each 'font' is really the same graphs with different codes so that it appears that you use different keystrokes for many of the same characters. You cannot change text from one DIS©93 font to another. The point of this is to allow people choice among three different keyboard layouts. *OldSyl* is similar to the Selectric. *LogiSyl* is a revision of this which lines up almost all the finals with their larger counterparts and groups as many of the characters as possible

in vertical columns, replacing the comma, period and slash keys with ᓐ ᓕ ᒃ respectively. *QalluSyl* lines up the columns over corresponding roman keys so that *ri, ru, ra, r* are in the *r* column (but *vi, vu, va, u* are elsewhere). This makes it easy to learn to type syllabics if you are already familiar with the English keyboard. With all three layouts you get dots in the same way. Press the ` key at top left before the taller characters and press the ~ (shifted `) before the shorter ones. The dot characters have ultra-positive width and so appear over the characters which follow.'The entire syllabic character set can now be typed without using the CTRL, ALT or FUNCTION keys!'[25]

For the two pre-Windows technological stages it was also necessary to develop the means of accessing the upper character set through the keyboard as well as the means of getting a printer to produce syllabics. These means are often software specific and printer specific. That is, a change in software or printer will often entail a change in the means of accessing and printing syllabics. This involves extra programming work and hence extra costs. These criticisms or weaknesses are getting less relevant as the industry gravitates to Windows as the standard DOS shell. It may be partly because of the diversity historically inherent in MS DOS solutions that none have been more widely used. With the Mac there is less diversity because the font technology is built into the operating system rather than added on.

MACINTOSH COMPUTERS AND SYLLABICS: MACTITUT

To my knowledge, only one microcomputer software package has adopted the interim working convention and enjoyed a degree of distribution. MacTitut was designed in Yellowknife and is currently in use by Language Bureau staff and others. The main user has historically probably been the Language Bureau but currently use has fallen off and the Naamajut font (and character set, see below: The Mac Option-Key Strategy; Outline Fonts) now predominates.

MacTitut consists of two parts. There is a keyboard translator file which has to be kept in the system folder.[26] This is best activated[27] by typing a control code, Option-Shift-I (for Inuktitut) and deactivated with OptionShift-E (for English). When activated it translates the signal sent from the keyboard to the higher code number corresponding to a syllabics graph. Normally when one types an *f*, ASCII 102, on the keyboard, one sees an *f* on the screen and expects an *f* to print on paper. But when MacTitut's keyboard translator is activated the keyboard scan code gets ultimately interpreted, not as 102, but as 163. In many Macintosh fonts 163 is the English pound sign £ but in the MacTitut character set it is the Syllable *ku* ᑯ. The second part of the MacTitut package consists of four bitmapped fonts, *Inuktitut, Kimmirut, Nunavut* and *Iqaluit*. These are installed into the system file with Font/DA Mover in the same way that one installs any font or Desk Accessory.

HISTORY OF MACTITUT

The initial version of MacTitut, 1.0, from 1984 was the result of a cooperative effort between Betty Harnum, then the Inuktitut Linguist in the Language Bureau, and two consultants working with NorDat/Microage, Steve Rose and Chris Belanger. Because its development preceded the computer workshop, its syllabics code assignments differed from those of the interim convention (see above: The Quest for a 256 Code Standard). Betty designed the bitmaps for the font called *Inuktitut*. The keyboard layout followed the one used then on the IBM Selectric typewriter. It included the apostrophe key (left of Return) as a *dead key*. To get a long vowel one pressed the dead key and then the syllable key. This mimicked the process on the Selectric and so involved no learning for those

who could already type. The shifted dead key (quotation mark key) produced the superscript circle ° for the */ai/* vowel sequence.[28]

Around 1987 version 2.0 was released. This used the codes of the interim working convention. Version 3.0 was released in 1990. It was necessary because earlier versions were incompatible with the Macintosh system updates.

OUTLOOK FOR MACTITUT

MacTitut has from the beginning competed with other Inuktitut-Macintosh word processing systems. It is the only one whose codes follow the interim working convention of 1985. It has not become the dominant package probably mostly because of its significant cost. There are other solutions which are free. There are also technical problems[29] with the program the most distressing of which is its tendency to cause crashes. It does not work at all with System 7 and new Macs now require System 7.

Until 1991 I thought that the problems faced by MacTitut might be experienced by any software package which applied the Superbrain ASCII code standard to the Macintosh. I thought it would have to use the same strategy which means putting a keyboard remapping file (init) in the System Folder. But this is not true. Using the free Mac developers' program, *ResEdit*, one can easily build a new keyboard layout to be used by the System. The French Canadian System allows one to switch between two such keymaps, *US* and *Canada Français*. This is done in the Control Panel called Keyboard under the Apple Menu. The keymap can be changed at any time and it is not neccessary to restart the Mac for the change to take effect. However, ResEdit cannot currently be used to make a keyboard driver which perfectly mimics what Mactitut does with syllabics. For the dotted characters, 45 dead key pairs using apostrophe as the dead key are required. ResEdit provides for only 31 pairs. But a mimic may not be necessary. Most users now employ the option keys (discussed below) to access the dotted characters.

THE MACINTOSH OPTION-KEY STRATEGY

Both Macintosh and DOS machines have 47 keys which when pressed produce a character. These can be combined with the Shift key to give access to a total of 94 characters, including upper and lower case letters, numerals, punctuation and a few other symbols. The Macintosh further doubles this total to 188 characters through the use of the Option key. For instance, *a*+Shift=*A*, *a*+Option=*å* and *a*+Shift+Option=*Å*.[30] There are ways of accessing codes beyond the basic 94 on a DOS machine but they have not been nearly as simple and convenient.[31] With the exception of MacTitut fonts and *Pang* (see immediately below), as far as I can tell, all Macintosh Inuktitut syllabics font developers have used the Option-key strategy. The best known of these may be *Naamajut, Iqaluit+, Nunavik, Umiujaq* and *Emilia*. The strategy is rather simple. Pressing *w,s,x* will give ᐃ,ᐅ,ᐊ and pressing *W,S,X* will give ᐱ,ᐳ,ᐸ. Then, to obtain syllables with the superscript dot showing double vowel one presses the same keys together with the Option key. That is, pressing Option-w,s,x gives ᐄ,ᐆ,ᐋ and Option-W,S,X gives ᐰ,ᐴ,ᐹ.

Because the option-key solutions all rely on the US keyboard layout, they encountered difficulty when System 7.0 was released because Macintosh changed the US layout for the new System. Some characters were no longer accessible in the same way. For these fonts I had earlier made a keyboard layout called *No dead key US* by removing the dead keys from the System 6 US layout. This simplified typing. For instance, ᒻ with the usual system 6 driver is obtained by typing option-*u* twice, but with *No dead key US* one types just option-*u*. The modified layout also works fine under System 7. I renamed it *Naamajut key* and sent a copy to Peter Stuemple for distribution with Naamajut for

System 7.0 users. System 7.1 now comes with an additional layout called *U.S. - System 6* which will give option-key users their familiar keystrokes back, including option-*u* + option-*u* for ᒻ.

THE MACINTOSH OVERSTRIKE STRATEGY

With Macintosh fonts it has been easy to design characters that appear above or below the character typed before it. This is called an overstrike. A font named Pang, developed by Jack Hicks, uses this approach to get the superscript dot. After typing ᐃ one presses the apostrophe key to get ᐄ. The superscript dot is a separate character with zero width that in effect backspaces and appears over the character entered before it. Other developers do not use this strategy for esthetic reasons. It is preferable to have a font in which the placement of the dot varies with the character. I have seen Pang only at the Rankin Inlet Language Bureau although I assume the name is short for Pangnirtung, the hamlet on the Cumberland Peninsula on Baffin Island, and this suggests that the font was originally used in that region.

BITMAPPED AND OUTLINE PRINTER FONTS

There are two kinds of syllabics printer fonts available for the Macintosh. These are named after the way the printer gets its information on where to place the dots on the page. A bitmap font is printed from the same information as the screen bitmaps. The pattern of pixels on the screen is more or less[32] sent to the printer where it is converted to dots of ink. In some point sizes, especially the larger ones, the letters will come out with jagged edges, even on a laser printer.

The outline[33] type of printer font is newer. It is designed for the more advanced laser printers. Outline fonts get their information from a mathematical description or formula which one can think of as an expandable tracing of a letter. The laser printer places dots inside the tracing or outline of the character. Outline fonts produce very high quality output. With MS Word 5.0, for instance, you can produce nearly perfect letters without jagged edges in any point size from 4 (1/18 inch) to 16,383 (227 inches).

THE BITMAPPED SYLLABICS FONTS

There were probably many pioneer developers of Macintosh syllabics fonts but the one who seems to be best known in the NWT is Ed Horne. He is apparently responsible for three typefaces called *Iqaluit, Nunavut* and *Kimmirut.* These are still used with the Option-key strategy as originally intended but they have also been incorporated into the MacTitut package. The MacTitut developers took the bitmaps, reassigned the codes to follow the interim working convention, and included them on the disk along with Betty Harnum's Inuktitut face. This modification has created some confusion. Two people with the Iqaluit font cannot exchange documents if one is created without MacTitut and one is made with it. This certainly confused me for a while as I was hearing different things about what I thought was the same 'font'. The confusion continues in 1993. There are also bitmaps called *Iqaluit+* and *Iqaluit** based on the Baffin Iqaluit font codes.

I know of three other syllabics fonts that rely on bitmaps for printing. One is Jack Hick's Pang font mentioned above (The Macintosh Overstrike Strategy) which uses an overstrike rather than an Option-key strategy. Two further bitmapped syllabics fonts called Umiujaq and Ilisautik[34] use the Option-key strategy. Donald Turcotte of St. André-Avellin developed Umiujaq but I have not been able to learn who made Ilisautik. There are quite possibly several other bitmapped fonts for syllabics in existence.

OUTLINE FONTS

I currently know of six outline fonts for syllabics. The three that are commercially available are called *Naamajut, Nunavik,* and *Emilia.* Naamajut is available through the Baffin Divisional Board of Education, Iqaluit. It was originally produced on contract by an outside firm and then significantly improved by Del Carry. Peter Stuemple is now in charge of development and distribution. The Nunavik font is distributed by the Makivik Corporation, Inukjuaq. I have seen this advertised in either MacWorld or MacUser.[35] The developer is François Girard of Montreal. The Emilia font is produced by Eiko Emori Inc., Ottawa. It is used in the January 1991 issue of Inuktitut Magazine. A fourth font called EskiLaser was designed by Robert Bryce of Saskatoon and is available as shareware. A fifth was developed in 1986 by Donald Turcotte in St. André-Avellin on contract for the Kujjuaq Health and Social Services Council.[36] This early outline font is not proportionally spaced. It was designed to be used with the Umiujaq bitmaps and bears the same name. I assume it has been supplanted in usage by the Nunavik font.

A sixth set of outlines was designed by Nelson Ruest, formerly of Resolutions, Vancouver, for use in the production of the GNWT Main Estimates. The final document was produced on a DOS machine through Ventura Publisher, a desktop publishing package. The outlines are in DOS format although they were originally made on a Mac. They are free to any user at this point. I have been unable to convert them back to Macintosh format. If this could be done, the outlines should be made available at no cost for the purpose of promoting and developing the language[37] on that of the IBM or Xerox typewriter and so have much in common.[38] But there is enough difference to cause significant difffficulty when a user moves from one font or strategy to another. Pressing the equals sign, =, on the top right of the keyboard gives ᐧ in Naamajut, ᐃ in Iqaluit (Bafffin, not MacTitut codes), and ᒻ in Emilia and MacTitut. To get ᐄ, one presses Option-*w* with the fonts that use an Option-key strategy, or *w* then ‘ (apostrophe) with Pang, or ‘ then *w* with MacTitut.

But now that I see how easy it is to develop keyboard drivers on both Mac and Windows, a standardized layout is no longer a critical issue. With the right driver, any layout can probably be used with any character set. When the 65K set becomes accessible, all syllabics users will likely employ the Unicode standard syllabics codes but there will be a variety of layouts and each user will be able to press the keys in the fashion with which they are familiar.

RECENT LANGUAGE BUREAU DEVELOPMENTS

It has been clear that for various syllabics computing needs the available solutions have been inadequate. Specifically, none of the character sets is adequate. It is possible and not too difficult to convert the Mac interface so that it is essentially entirely in syllabics and can be used by unilingual Inuit but a special, well thought-out character set is required for the conversion. The same is true for a program which converts between roman and syllabics. Dictionary and terminology work can be improved with a different character set. Until the 65K set becomes available, it is useful to have something in the interim which will facilitate work by unilingual Inuit and scientific projects. To this end, I developed a new character set in consultation with Dirmid Collis, a linguist currently resident in Winnipeg, and David Moffat, a Mac language software developer from Chapel Hill, North Carolina.

The new character set is called *Inuit ASCII.* The intention was originally to leave the lower 128 codes unscathed and to restrict all the special characters to the upper 128. In this, it would resemble the Mactitut set. Two exceptions had to be made to this princi-

ple, because of a shortage of space as will become clear. Inuit ASCII also is a Macintosh-specific character set in that it recognizes the upper ASCII characters which the Mac gives special functions to, such as the ellipsis (... 201), the non breaking space (202), the n- and m-dashes (– 208, — 209), the printers' quotes (" " ' ' 210-13), and the copyright (© 169) and trademark (™ 170) symbols. It also keeps recognizable upper ASCII symbols where the Mac has them. This applies to *ñ* (150) and *Ñ* (132) which are necessary to write Uummaqmiutun (Inuit language from Aklavik and Inuvik).

Besides trying to keep the ASCII and Mac character sets as intact as possible, I also tried to include all the characters needed for computer processing of Inuit language in the NWT. This means that besides standard roman characters contained in the lower ASCII, and besides the standard 59 syllabics characters in current use, two other types of characters would have to be included. One type consists of the special roman letters used to write Inuit language. Many dialects have a voiceless-*l* so a barred-*l* symbol, *ɬ*, is needed. Uummaqmiutun requires two additional symbols, the palatal nasal *ñ* mentioned above, and the *r*-with-a-hat or *r*- circumflex, *r̂*, which was developed to stand for a voiced retroflex phoneme since the letter *r* was already used to write the voiced uvular fricative *[ʁ]*. Upper case variants of these are also required so that six additional roman letters are needed.

The second type of character which needs to be included is a set of syllabics symbols for the voiced retroflex phoneme of Natsilingmiutut. Currently the phonemes /y/ and /r̂/ are not distinguished, both being written with the /y/ or 'j' symbols. Linguistic work, and especially dictionary work, must describe these phonemes. There have been some efforts made by the Natsilik themselves to distinguish these sounds in writing. It is possible that next fiscal year the Language Bureau may be able to run some kind of public process in Taloyoak (formerly Spence bay), Pelly Bay and Gjoa Haven, the Natsilik hamlets, for the people to decide whether they should distinguish these sounds and if so, through what set of symbols. In the meantime, a set of seven computer codes in InuitASCII has been reserved for a /r̂/ row in syllabics. The characters assigned to these codes can change shape later if need be. Tentatively, for Dirmid Collis's Natsilingmiutut dictionary work, the /r̂/ symbols are given as underlined /y/ symbols. That is, besides regular ᔾ ji, ᔿ jii, ᕀ ju, ᕁ juu, ᕃ ja, ᕄ jaa, ᕆ j, there are underlined ᒃ r̂i, ᒄ r̂ii, ᒅ r̂u, ᒆ r̂uu, ᒇ r̂a, ᒈ r̂aa, ᒉ r̂. In essence the underlining is a diacritic distinguishing one use of a set of symbols from another. It is perhaps the least obtrusive way of noting the linguistically relevant contrast.

The accompanying chart (next page) gives the decimal codes for Inuit ASCII. There are darker borders around the two non-standard characters in the lower ASCII. There are double borders around the Mac standard upper ASCII which are retained. The non-standard roman in the upper ASCII are shaded. It will be seen that the rest of the characters in the upper ASCII are syllabics and that there are no empty spaces.

Because of the requirements for this character set, and the shortage of space, it was necessary to place two non-standard roman characters in the lower ASCII. This is the least undesirable way of accomplishing the goals. The barred-*l*, *ɬ*, replaces tilde, ~, and the *r*-circumflex, *r̂*, replaces carat ^. The characters ~ and ^ are little used in documents, probably not at all by most people, and the ~ mnemonically invokes *ɬ*, while the ^ reminds one of *r̂*.

Inuit Ascii Decimal Codes

	0	1	2	3	4	5	6	7	8	9	
30			space	!	"	#	$	%	&	'	30
40	(	)	*	+	,	-	.	/	0	1	40
50	2	3	4	5	6	7	8	9	:	;	50
60	<	=	>	?	@	A	B	C	D	E	60
70	F	G	H	I	J	K	L	M	N	O	70
80	P	Q	R	S	T	U	V	W	X	Y	80
90	Z	[	\	]	ˆ	_	`	a	b	c	90
100	d	e	f	g	h	i	j	k	l	m	100
110	n	o	p	q	r	s	t	u	v	w	110
120	x	y	z	{	|	}	˜	delete	Δ	Δ	120
130	ᐅ	ᐅ	Ñ	ᐊ	ᐊ	ᐱ	ᐱ	ᐳ	ᐳ	ᐸ	130
140	ᐸ	ᐸ	ᑎ	ᑎ	ᑐ	ᑐ	ᑕ	ᑕ	ᑦ	ᑭ	140
150	ñ	ᑭ	ᑯ	ᑯ	ᑲ	ᑲ	ᑳ	ᑫ	ᑫ	ᒍ	150
160	ᒍ	ᒐ	ᒐ	ᒉ	Ł	•	¶	ß	®	©	160
170	™	ᒥ	ᒥ	ᒧ	ᒧ	ᒪ	ᒪ	ᒫ	ᓂ	ᓂ	170
180	ᓇ	ᓇ	ᓈ	ᓈ	ᓉ	ᓕ	ᓕ	ᓕ	ᓕ	ᓗ	180
190	ᓗ	ᓘ	ᓚ	ᓚ	ᓛ	ᓛ	ᓭ	ᓭ	ᓯ	ᓱ	190
200	ᓱ	...	non-breaking space	ᓴ	ᓴ	ᔦ	ᔦ	ᔨ	–	—	200
210	"	"	'	'	ᔭ	ᔭ	ᔾ	ᔾ	ᕆ	ᕆ	210
220	ᕈ	ᕐ	ᕐ	ᕓ	ᕓ	ᕖ	ᕖ	ᕗ	ᕙ	ᕙ	220
230	ᕚ	ᕚ	ᕝ	ᕝ	ᕞ	ᕠ	ᕠ	ᕡ	ᕡ	ᕢ	230
240	ᕢ	ᕣ	ᕤ	ᕤ	ᕥ	ᕥ	ᕦ	ᕦ	ᕧ	ᕭ	240
250	ᕭ	ᕮ	ᕯ	ᕱ	ᕱ	ᕲ					250
	0	1	2	3	4	5	6	7	8	9	

One of the objectives of this set is to develop an Inuit syllabics Macintosh interface. For this to work, we need to retain certain characters which are used by the system or are commonly used by software. For instance, if you check under the Apple menu, in the first item, **About (the current program)...,** you see the ellipsis character, the three dots. If you open this item, you will see information about the currently running program. All this information is in characters from the lower 128 codes except the copyright symbol (©) (dec. 169). These kinds of messages will remain unchanged in a syllabics operating system because none of the characters, including ... or © have been replaced. Probably the only thing which would appear in syllabics would be the word **"About"** which occurs in the menu.

TEXT OASIS

We now have easy to use, inexpensive software for converting from roman to syllabics and vice versa. This uses an off-the-shelf desk-accessory called Text Oasis made

by David Moffat of Chapel Hill, North Carolina. David originally designed it work with the Russian language. He desired to transliterate roman to Cyrillic (Russian, Bulgarian, etc.) writing, Cyrillic to roman, and to transliterate among the Russian character sets of which there are now more than forty. Text Oasis is now used for several other languages including Inuit.

Along with Text Oasis you need a conversion table for every desired conversion. I have written so far eight conversion tables, listed here in the same alphabetic order they appear in Text Oasis (IA abbreviates InuitASCII).

—IA Syllabics to Roman
—Iqaluit+ to InuitASCII
—Iqaluit+ to Naamajut
—MacTitut to InuitASCII
—MacTitut to Naamajut
—Naamajut to InuitASCII
—Roman to IA Syllabics
—Umiujaq to InuitASCII

These tables do three kinds of conversion. One is conversion between roman and syllabics. This requires the Inuit ASCII character set as can be seen in the titles "IA Syllabics to Roman" and "Roman to IA Syllabics". The second function is to convert from lesser used sets into more prevalent ones. Thus one can convert from Iqaluit+ or Mactitut characters into Naamajut which is currently the most widely spread set (and font) in the NWT. The third function is to convert from the syllabics sets which I have access to (Iqaluit+, Naamajut, MacTitut, Umiujaq), to Inuit ASCII.

I used Text Oasis in 1992 to prepare materials for a terminology workshop in Cambridge Bay. These were to show the participants what other dialects had come up with for the target terms. In the sample below, the second column was converted from a Language Bureau MacTitut document to Inuit ASCII. This was then converted to roman and pasted in the third column. The entry in the fifth was taken from the Arctic College terminology list (1990) and typed by hand in roman. This column was then converted into syllabics and pasted in the fourth column.

fingertip	ᐳᔪᐊᐢ, ᒪᐃᓐᐃᐢ	pugjuit, mainait	ᐳᔪᐊᐢ, ᒪᑎᓐᐃᐢ	pujjuit, maunait

Dirmid Collis has used Text Oasis and the Inuit conversion tables in his Natsilingmiutut dictionary work. Originally his computer files were done in roman writing. Text Oasis provided the means to convert the Inuit language material to syllabics. The dictionary can now be checked by a native speaker who is literate in syllabics, and eventually it can also become a reference tool for syllabics users.

I have further had one inquiry from Baker Lake and one from Coppermine about Text Oasis. These people would like to enter text in roman on a computer but be able to produce documents in syllabics. There are bilingual Inuit people who can type in English but not in syllabics. Text Oasis makes learning a second way of typing unnecessary.

The Text Oasis program can be obtained from David W. Moffat, glps Products, PO Box 3454, Chapel Hill, NC 27515; Tel. (919) 968-6780, AppleLink D2353@applelink.apple.com. The price a few months ago was $50 but David was considering an increase. The Inuit Conversions file, used in conjunction with Text Oasis (and useless without it!), can be had at no cost from Doug Hitch, Language Bureau, GNWT, PO Box 1320, Yellowknife, Northwest Territories X1A 2L9; Tel. (403) 920-6353, FAX (403) 873-0107.

SYLLABICS DICTIONARIES

The Inuit ASCII character set will allow us to easily produce wordlists in syllabics, in the order of the syllabary. There is currently no program which will sort syllabics. Basic sorting is of course a simple software problem but the fact remains that until recently no one had accomplished it, even in a klutzy way. The Language Bureau sponsored development of the Mac software to allow this kind of sorting in the most elegant way possible — at the system level. This involved rewriting the itl2 system resource .

Stefan Embleton of SunValley Software in Winnipeg modified the itl2 to properly sort the Inuit ASCII character set. A Control Panel lets you switch between *US* and *Inuit* tables in the itl2 on the fly. For System 6 the Control Panel is called *International.* For System 7 it is called *Text.*

SYLLABICS OPERATING SYSTEM

The production of a new itl2 for Inuit syllabics will make it possible to have the Macintosh user interface entirely in syllabics. That is, all of the menu names and items, and all of the file names can be in syllabics. Instead of seeing **File, Edit,** etc. at the top of the screen, the appropriate corresponding Inuktitut words can appear there. Within folders, document names can be in syllabics and appear in syllabics alphabetic order automatically. Adapting the Macintosh entirely to syllabics will make it easier for unilingual elders to use the machine, and it will encourage Inuit children to value their language more. Dirmid Collis of Winnipeg and Allan Angmarlik of the Baffin Divisional Board of Education are collaborating on this at the moment.

ITL2 AND OFF-THE-SHELF SOFTWARE

Any software that uses the system to do sorting can be used build dictionaries, terminology lists or vocabularies. This can be a real boon to language workers. For instance, because MSWord 5 uses the itl2 to sort, it can be used by Inuit curriculum developers to produce alphabetized word lists. By using the indexing function in Word 5, one could create the index to a book or manual in syllabics. By extending Word's indexing function a little, it could be used to create a list of all the words in a text with the page numbers for all occurrences.

Probably a better tool for this kind of task is Vocabulary Creator™, from glps Products of Chapel Hill. David Moffat designed this to work with the itl2. From a unilingual or bilingual text, Vocabulary Creator can create a list of all the words, sorted in alphabetical order. Then it allows you to add definitions automatically from a pre-existing definitions file. Words which are not yet entered in the definitions file can be identified at this point and entered. As time goes on the definitions file increases in size. It gets better able to gloss texts and could eventually become the basis of a dictionary or spell-checker. However, for languages with highly complex morphology, like Inuit or Athapaskan, no matter how long your word list or definitions file, there will always be morpheme combinations (words) which are not listed. Spell-checkers cannot work unless they are able to analyze a word into morphemes.

Apparently 4D, the advanced data-base program for the Mac, also recognizes the itl2. This program has then tremendous potential for dictionary work. 4D is challenging to set up for a particular task but after it is customized it can be used by language workers with average computing skills.

Notes

1. An earlier version of this paper completed in the Fall of 1990 was entitled *Syllabics, Inuktitut, Microcomputers and Ascii coa'es.* In April 1991 it was revised and given the present title. When META asked for a laser printout, February 10, 1993, I made further revisions. The technology in this area develops so quickly that anything described as current here may be outdated by the time the article appears.
2. In researching this paper I spoke to many people. I would like to express my gratitude to Betty Harnum, Mary Pepper, S.T. (Mick) Mallon, Bob Johnson, Chris Belanger, John Veringa, Nelson Ruest, Del Carry, Andrew Tagak, Susan Sammons, Mike Shouldice, Louise Campbell, Peter Royle, François Girard, Donald Turcotte, Dirmid Collis, David Moffat, the Eiko Emori company, Dirk Vermeulen, Bob Bryce, Ed Peters, Peter Stuemple, Stefan Embleton, Peter Baril and Rici Lake. My apologies to anyone whom I have forgotten to mention and to anyone who would like to have been contacted. I bear complete responsibility for any errors or inadequacies here.
3. This document does not represent the official views of the Language Bureau, the Department Responsible for Education, Culture, and Employment Programmes or the Government of the Northwest Territories.
4. All code numbers are given in decimal. Developers generally prefer hexadecimal.
5. CSA CSIC 1992.11.19, p. l.
6 Esther Wesley, Project Manager, National Association of Cultural Education Centres, Ojibway Cree Cultural Centre, 152 Third Avenue, Timmins, Ontario, P4N 1C6.
7. Paul Green, Technical Advisor, NACEC, c/o First Nations Technical Institute, R.R. #1, Deseronto, Ontario, KOK 1XO.
8. Andy Kwan, Federal Working Group, Communications Canada, 3701 Carling Ave, P.O. Box 11490 Station H, Ottawa, Ontario, K2H 8S2.
9. Nortext 1985.
10. Nortext 1985, preface.
11. Nortext 1985, p. 8.
12. Johnson 1985.
13. The 105 symbols consist of 15 rows x 3 columns x 2 (single and double vowels) + 14 finals + *H*. Bob Johnson's Inuktitut set does not assign separate codes to the digraphs *qi* ᖿ, *qu* ᖽ, *qa* ᖾ, *q* ᖻ, *ngi* ᙆ, *ngu* ᙇ, *nga* ᙈ. It includes an extra *i* column with a superscript circle indicating */ai/*. His set totals 106: 13 rows x 3 columns x 2 (single and double vowels) + 13 */ai/* Syllables + 13 finals + final *ng* + *H*.
14. Since the traditional syllabary had 12 rows, inverted *qi, ngi,* and *li* symbols might not be necessary.
15. Johnson 1985, tables 6-8.
16. Johnson 1985, p. 5-7.
17. Report on the Syllabic Computer Workshop, op. cit., Appendix 3, p. 27
18. 96 in the lower range and 128 in the upper.
19. Nortext 1985, p. 8.
20. Johnson 1985, p. 3 = Nortext 1985, p. 23.
21. The Language Bureau now has the ability to sort syllabics on the Mac. See below: Syllabics Dictionaries; it12 and Off-the-Shelf Software.
22. Vermeulen 1989, Appendix 1.
23. Robert Slavin, Government Services, Yellowknife some years ago was developing a syllabics solution for WordPerfect. But WordPerfect changed before the project was completed. He probably would have used a stage two strategy at that time.
24. Many DOS programs use one or more of the upper 128 codes as control codes so Bob has to adjust his syllabics work accordingly.
25. Undated DIS©93 pamphlet, Datarctic, Iqaluit.
26. In Mac terminology this kind of program is called an *init* which stands for *initialization* file. In DOS it may be called a TSR 'Terminate and Stay Resident'. In a way it is lurking in the background waiting to spring into action when the correct combination of keystrokes is entered.
27. The developers intended that the keyboard remapping be automatically invoked when an Inuktitut font was selected. However this does not work well with some programs such as MS Word.
28. The Inuktitut sequence of vowels */a/+/i/* was originally written with the syllables borrowed from Cree used for the Cree long vowel */e:/*. The other Inuktitut vowel sequences */au, ui, ua, iu, ia/* have been written with two syllables from the start. The Cree *e* syllables, although providing a shortcut for writing */ai/,* violate the overall orthographic pattern for vowel sequences. They are redundant. The superscript circle converts the *i* syllables into *ai* ones, mimicking the effect of the fourth column. It is also redundant. The ICI standard orthography of 1976 eliminated the fourth column but retained the circle diacritic. This was done to permit typesetters the option of using one character instead of two. Also, there appears to have been mechanical motivation related to the development of the syllabic element for the IBM Selectric typewriter: '... one set of redundant syllabic symbols was omitted from the ICI orthography not only for logical reasons, but also in order to free more keys on the keyboard' (Mallon 1985). The circle is not used today in official docu-

ments. It is not included in any of the outline laser fonts currently available.

29. Besides frequent system hangs or crashes a few other problems can be reported. It is not possible to work on some characters in the program Fontographer 3.0. Somehow MacTitut blocks access to them. Also, use of any fonts made with Fontographer will crash the System if MacTitut is still in the System Folder. When the Inuktitut keymapping is activated it is not possible to use the Command keys such as Command-S for Save or Command-Q for Quit. One also cannot enter information in Dialog Boxes such as when giving a title to a document when saving it for the first time.
30. The Caps Lock key can also be programmed separately to double the 188 to 376 which exceeds the number of characters available in the 256 code set (223 characters in the Mac set + 32 control characters [0-31] + 1 'delete' code [127]). Extended keyboards have a Control key which can also be used in any combination with character, Shift, Caps Lock and Option giving a total range of 752 combinations.
31. Dos, more or less the IBM equivalent to the Macintosh System, allows one to enter any code by holding down the Alt key, entering the code number on the numeric keypad and then releasing the Alt key. Macro or TSR software can provide shortcuts but these are all external to the operating system, not internal as on the Macintosh. With Windows it is likely that the keyboard will become as useful for accessing special characters as the Mac is, if it is not at that stage already.
32. Printer drivers will often locate a larger bitmap if it is available and use it to make better looking characters. For example, with an ImageWriter II, to print 12 point characters the best results come with 24 point bitmaps installed in the system.
33. Outline fonts are frequently referred to as *PostScript* after the brand name used by the Adobe company for its outline formula which has been the de facto microcomputing standard. Apple and Microsoft have also produced an outline formula called *TrueType*. On the Mac it works with System 6.0.7 with the TrueType init in the System Folder and it is built in to System 7. TrueType also renders screen images of characters at almost any point size. The same thing is done by Adobe Type Manager for PostScript. TrueType produces amazing output on an ImageWriter of all things. Language Bureau Dene staff members in the regions are now producing high quality documents with only a Mac Plus and an ImageWriter II but using TrueType. Apparently Adobe Type Manager and PostScript outlines will also give high quality output on an ImageWriter II.
34. Both have come onto my desk through Kuputiriji, a program formerly distributed by ICI which was intended to transliterate automatically from syllabics to roman and vice versa. 'A suitcase desk accessory to transcribe Inuit syllabics to Roman letters or the reverse. For use on Macintosh +, SE, II, IIx and 30 computers' (from the label on *Kuputiriji*, Inuit Cultural Institute). It does not seem to work properly. For instance, it will convert Roman *inuktitut* to Syllabics ᐃᓄᖅᑕᑦ *(iruktitut)*. The Language Bureau, Dirmid Collis of Winnipeg and David Moffat of Chapel Hill, North Carolina, have together produced a similar and working desk accessory arrangement for Inuit syllabics (described below: Text Oasis).
35. Thanks to Nunda Rao of Government Services, Yellowknife for pointing this out.
36. Conseil de la Santé et des Services Sociaux.
37. Peter Baril of Datarctic may be able to help with a DOS program called FontMonger. I am sending him a disk and should know shortly if this will work.
38. The exceptions to this rule now are QalluSyl and LogiSyl described above; see: DOS Computers and Syllabics.

BIBLIOGRAPHY

BURNABY, Barbara (Ed.) (1985): *Promoting Native Writing Systems in Canada,* Toronto. OISE Press/The Ontario Institute for Studies in Education.

JOHNSON, Bob (1985): *A Proposal for Computer Encoding Of Syllabics Text,* Appendix 3 in Nortext 1985, p. 21-32; also Appendix 2, in Vermeulen 1989.

MALLON, S.T. (1985): "Six Years Later: The ICI Dual Orthography for Inuktitut, 1976-1982", *Burnaby* 1985, p. 137-157.

Nortext (1985): "Report on the Syllabic Computer Workshop, Held by the Kativik School Board, Montreal, March 14 to 16, 1985", Unpublished circular prepared by Nortext Information Design Ltd., Nepean, Ontario.

VERMEULEN, Dirk (1989): *Report on the Native Language Communications Survey.* Government of Canada Department of Communications, contract # CRC 8-3604. Vermeulen Studios, Beamsville.