4 March 2009

Technical Specification

Version 3.1.4

Global Keyboard Optimised for Small Wireless Devices (GKOS)

GKOS Project

Functional Requirements

THIS SPECIFICATION IS MAINTAINED BY SEPPO TIAINEN (GKOS@GKOS.COM) WHO ALSO HOLDS THE COPYRIGHT FOR IT.

CONTENTS

Foreword....2

2 Keyboard physical layouts....5

4 GKOS Modes and special functions....9

5. The GKOS Keyboard Scanning (Polling)....10
A. Basic Scanning....10
B. Intelligent Scanning - The recommended way....10
C. Teach mode....10
D. Special mode....11
E. Game and pointer control....11

6. Document history....11

ANNEX 1 - How to type on GKOS....12

ANNEX 2 – Character sets and virtual keyboards....13
1 The complete character set....13
2 The national character sets....14
3 Other virtual keyboards....15

ANNEX 3 – Chordon Technique....17

Entering Chordons on a GKOS Keyboard....17

Foreword

I am proud to present in detail this new user interface for small mobile devices. Those who care to learn the GKOS alphabet can now enter a completely new dimension of messaging, namely, the ability to chat and send text messages at the same speed as with the PC - but while being mobile. No spectacles or magnifying glass. No need to keep on teaching your predictive phone new words. No separate special tools (stylus, pen) to loose. Back to using the little author in you in a creative way, and fast.

Despite the many operational advantages obtained, a device with a GKOS keypad can be made small and solid, does not look too peculiar to show in public, its use is discrete, unambiguos, fast and does not require any preparations.

There is no doubt that speech and text messaging are the only real killer applications of mobile systems today. Not even the present clumsy text entry methods and fuzzy small displays with tiny characters have been able stop the SMS from getting popular. The good news is that now those last remaining drawbacks of mobile text entry can be turned into things of the past.

Seppo Tiainen
Veikkola, FINLAND

The Global Keyboard Optimised for Small Wireless devices (GKOS) was developed by Seppo Tiainen during years 1999 and 2000. It was first published on 5 October 2000 and further refinement of the concept has been done since that. Typing speeds of 250 characters per minute (50 WPM) have been achived with it (1 minute of typing ordinary text i.e. letters + punctuation). To find out the latest information on GKOS, visit the website at http://gkos.com

1 Objectives and Background

This GKOS specification has been prepared in order to define an open standard for a compact physical keyboard capable of providing all PC QWERTY features in a small size as well as providing fast text entry, dialling and pointer/game controls on wireless devices without taking up any space on the display or on the front panel.

One of the objectives is that the user can choose between different types and brands of small devices and still be able to type easily using the skill once obtained, as is the case with QWERTY when it comes to larger equipment.

The GKOS keypad is basically a set of 6 keys (3+3) on the back of the device. It is operated mainly by two hands. The characters on each 3-key set depend on the shift function pressed on the other 3-key set. Because the keyboard is only 3 fingertips wide, it fits even on the smallest of cellular terminals. It can also be used in many other types of applications that use cursor control and digit/text entry (remote controls, electronic notebooks, cameras etc.).

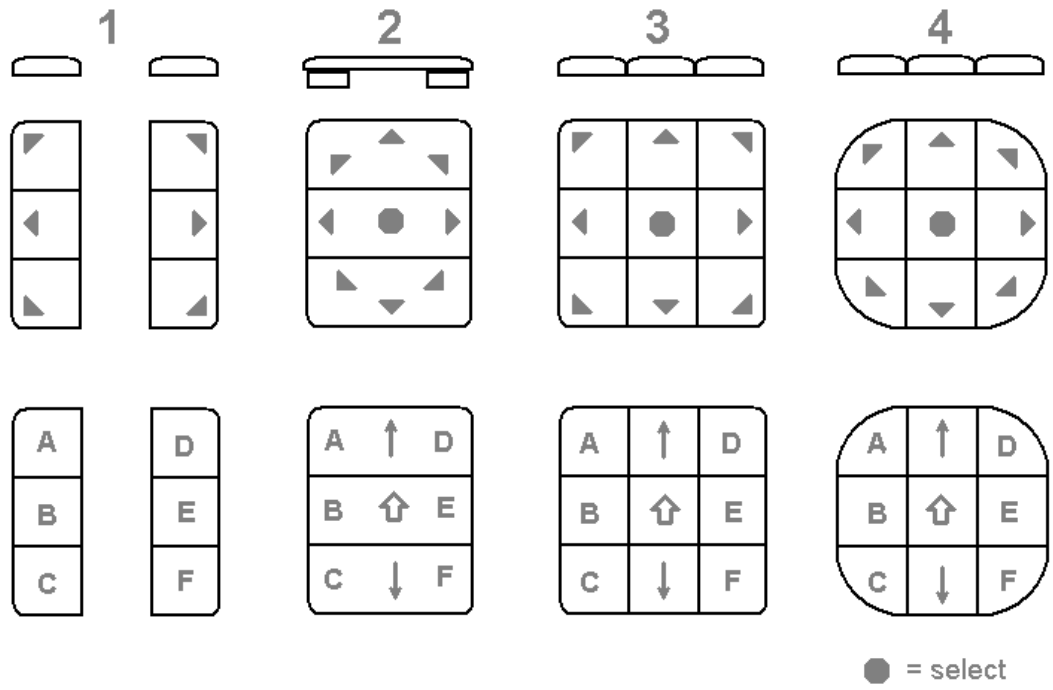
Because GKOS is a new method of typing there is a learning curve but, on the other hand, all these goals have been met:

1. The keyboard takes no space on the front panel or on the display area.
2. Low cost and easy to integrate on a small terminal.
3. The same hardware suits most languages.
4. Easy enough for the beginner (on-screen help to start with).
5. Suitable for fast typing (experienced user: 30-45 wpm, expert: 45-60 wpm, even faster with word shorcuts)
6. Does not require full attention of the typist (no table of characters to watch continuously etc.).
7. Usable also in the dark or with eyes closed.
8. There is no need to have backlight on the buttons (lower power consumption)
9. In addition to text entry, includes all functions of the PC QWERTY keyboard.
10. Can be used to control all functions of a mobile phone (dialling etc).
11. Operates as a game controller (or even as a pointer control in some cases).
12. Facilitates display browsing and menu selection.
13. No desk needed: can be used while standing, lying, walkinq etc.
14. Can also be used on a desk (e.g. the SixBack implementation)
15. Fully integrated and does not require separate tools.
16. Seamlessly combines with a mouse or other pointing device.
17. Usable as a hand held (remote) PC keyboard if desired (by those preferring the new method).
18. Does not require special physical finger skills (few and easy combinations per hand).
19. Treats left and right-handed people equally.
20. GKOS typing skill does not interfere with QWERTY typing skill (different enough).
21. Does not necessarily increase the physical size of the mobile phone/terminal.
22. A stand-alone GKOS keyboard can be extremely small (matchbox, pen...)
23. Can be combined with the conventional dialling pad (keys 1, 2 and 3 are then common) if desired.
24. The same virtual keyboard, if visible on the screen, can be used either by a stylus/pen or on the GKOS keys on the back of the same device.
25. Room for national characters reserved on equal basis compared to just A to Z.
26. No need to attach the device to the user (no belt, hand strap or alike required)
27. In certain implementations (8 or 9 keys), the keypad can be rotated by software to match the position (horisontal/vertical) of the terminal and screen aspect ratio used.
28. Open standard, free for anyone to use, for other purposes too.

See www.gkos.com for application ideas and more advantages.

2 Keyboard physical layouts

There are several physical layouts possible for the GKOS keypad. Some of them are shown below and their characteristic are described. They all can be used for game control, menu selection or for moving a pointer on the screen, as well as for ordinary typing. The symbols are shown as seen 'through the device' (left hand presses the keys on the left) separately for game control mode and typing mode. The game/pointer control symbols can be marked on the keys also in practice.



(1) The basic GKOS layout is two sets of 3 keys. The sets can be close to each other or completely separated, e.g. on the sides of a display back panel. This requires the use of both hands.

(2) This layout of only 3 keys combines the keypads in such a way that pressing in the middle of a combined key causes the switches at both ends of the key to be turned on. All characters can optionally be typed by single hand but fast typing is still done using both hands.

(3) By having nine independent keys gives one more advantage: the orientation of the keyboard can be controlled (rotated) by software according to the display aspect ratio in use (wide screen or upright position of the device). The three extra keys provide the same characters as the two keys on both sides of them pressed simultaneously. Single hand use is possible in many cases. Two hands are needed for typing more fluently.

(4) The rounded pad looks more like a pointer/game control. Otherwise it is similar to layout (3).

For each case above, a separate pointer control device and the related mouse buttons can exist in addition. They can be on the front panel and operated by thumbs, or the control can also be on the back panel, operated by index or middle finger. The pointer control may even be located in the middle of the GKOS keypad.

3 Character set definitions

The next three tables define the character set for keyboard implemention.

GKOS ref. num.	123 Mode preceded by:			ABC mode preceded by:			FUNCTION preceded by:	Key code
		SYMB	SHIFT or CAPS		SYMB	SHIFT or CAPS	Ctrl SYMB	
1	1	a	<1>	a	1	A	F1	1
2	2	b	<2>	b	2	B	F2	2
3	3	c	<3>	c	3	C	F3	4
4	4	d	<4>	d	4	D	F4	8
5	5	e	<5>	e	5	E	F5	16
6	6	f	<6>	f	6	F	F6	32
7	0	g	<0>	g	0	G	F10	24
8	7	h	<7>	h	7	H	F7	25
9	8	i	<8>	i	8	I	F8	26
10	9	j	<9>	j	9	J	F9	28
11	#	k	<#>	k	#	K	<F10>	48
12	@	l	<½>	l	½	L	F11	49
13	½	m	<&>	m	&	M	F12	50
14	&	n	<@>	n	@	N	F13	52
15	+	o	<+>	o	+	O	F20	3
16	%	p	<%>	p	%	P	F14	11
17	=	q	<=>	q	=	Q	F15	19
18	^	r	<¥>	r	^	R	F16	35
19	*	s	<*>	s	*	S	<F20>	6
20	$	t	<€>	t	€	T	F17	14
21	€	u	<$>	u	$	U	F18	22
22	£	v	<£>	v	£	V	F19	38
23	(	w	<(>	w	(	W	HELP	40
24	[	x	<[>	x	[	X	NUM LOCK	41
25	<	y	<<>	y	<	Y	SCROLL LOCK	42
26	{	z	<{>	z	{	Z	PRT SCREEN	44
NATIONAL LETTERS								
27 (2	)	Ü / TH	<)>	Ü / TH	)	Ü	BREAK	5
28 (2	]	Å / that_	<]>	Å / that_	]	Å	PAUSE	13
29 (2	>	Ä / the_	<>>	Ä / the_	>	Ä	SYS REQ	21
30 (2	}	Ö / of_	<}>	Ö / of_	}	Ö	CLEAR	37
PUNCTUATION								
31	.	:	<.>	.	:	.	<.>	34
32	,	;	<,>	,	;	,	<,>	20
33	!	|	<!>	!	|	!	<¡>	12
34	?	~	<?>	?	~	?	<¿>	33
35	-	_	<->	-	_	-	<->	17
36	’	”	<’>	’	”	’	<’>	10
INTONATION								
37	\	`	<\>	\ **or** `(1	`	\	<\>	51
38	/	´	</>	/ **or** ´(1	´	/	</>	30
39 (2	¨	|	<¨>	¨ **or** and_	|	¨	<¨>	53
40	¤	§	<~>	~	§	¤	<~>	46
41 (2	ˇ	<¤>	<^>	^ **or** to_	<¤>	ˇ	<^>	29

Table 1. GKOS Character codes. Note 1): Either accents or slashes (recommended) are primary depending on the national keyboard character layout. Typing the intonation character and a letter as a Chordon (Annex 3) always creates an intonated letter. Note 2): These symbols may vary in national layouts (the_ means 'the' plus a space). <> means: spare but may have a default value.

GKOS ref. number	All modes (see pointing mode in addition)	Key code
NAVIGATING		
42	Arrow Up	9
43	Arrow Down	36
44	Page Up	27
45	Page Down	54
46	Backspace	7
47	Arrow Left	15
48	Word Left	23
49	Home	39
50	Space	56
51	Arrow Right	57
52	Word Right	58
53	End	60
CONTROLS		
54	ENTER	59
55	TAB	61
56	ESC	31
57	DELETE	62
58	INSERT	43
59	SHIFT	18
60	SYMB	45
61	123-ABC	63
62	CTRL	47
63	ALT	55

Table 2. Functions common to all modes of Table 1.

The GKOS reference number is just a global reference to the key combination. Key Code is the mathematical value of the combination when keys A, B, C, D, E, and F are given values 1, 2, 4, 8, 16 and 32 respectively.

Note that these tables are not intended for learning to type on GKOS. It would be far too difficult this way. Simple illustrative virtual keyboards exist for that purpose (ANNEX 2).

SHIFT, ALT, CTRL and SYMB keys have an effect only on the first following character. Pressing SHIFT twice sets CAPS LOCK which can be turned off by one more SHIFT. Pressing ALT or CTRL sends immediately information that the key is being pressed (’make’ code) and the release of ALT/CTRL (’break’ code) is sent after another character is pressed or when pressing the same key again.

123-ABC changes the mode of the keyboard. Exiting the new mode is done only by pressing 123-ABC again. Pressing 123-ABC three times should always turn on the default mode of the keyboard (often ABC mode).

GKOS ref. number	Pointing mode	Key code
POINTER NAVIGATING		
42	Up	9
44	- fast	27
43	Down	36
45	- fast	54
1	Up Left	1
16	- very fast	11
15	- fast	3
2	Left	2
46	- fast	7
3	Down Left	4
22	- very fast	38
19	- fast	6
4	Up Right	8
8	- very fast	25
7	- fast	24
5	Right	16
50	- fast	56
6	Down Right	32
14	- very fast	52
11	- fast	48
POINTER ACTIONS		
59	Select	18
27	Left Click	5
54	(Left Click)	59
23	Right Click	40
56	(Right Click)	31

Table 3. Pointing mode for moving the pointer and selecting items (like links and menu items) on the screen. Same functions can be used in game controls.

The pointing mode is normally activated by the application but there is also a command to activate it: SHIFT followed by 123-ABC, then exit by 123-ABC.

4 GKOS Modes and special functions

Generally, the functions below can be selected on the application level by using menus that are easy to the user. Sometimes it may, however, be useful to be able to access these special features directly on the keyboard.

Modes

Modes are entered by the 123-ABC key that may be preceded by Ctrl, Alt, Shift or SYMB. To exit a mode, 123-ABC key is pressed (to exit Braille mode, press 3 times 123-ABC). 123-ABC always returns to ABC mode except, while in ABC mode, it activates 123 mode:

123-ABC	-> 123 mode
Alt 123-ABC	-> alternative mode*
Shift 123-ABC	-> pointer/game control mode ('GKOS mouse')**
SYMB 123-ABC	-> <reserved for e.g. Braille mode (raised dots code)>
Ctrl 123-ABC	-> Control mode

* This may include switching between national keyboard layouts, directly or based on a menu. For example, Alt 123-ABC can quickly switch between Finnish and English national character sets.

** Note that CAPS LOCK does not have an effect on the 123-ABC function but it must be turned off before entering Shift 123-ABC.

Special functions

Special functions can be activated by SYMB that may be preceded by Ctrl, Alt or Shift. If there is a need to return without selecting any function after SYMB, Esc can be pressed.

SYMB	Pick one symbol from 123 mode or shift a punctuation/special character
Alt SYMB	To use Alt for the next characer requiring SYMB
Shift SYMB	[Application menu appears]
Ctrl SYMB	Pick one Function key or one special key (F1...F12, PrtScr, Scroll, NumLock...)

SYMB Ctrl	To use SYMB for the next characer requiring Ctrl During remote control operation, SYMB Ctrl x... Enter changes the device to control, or opens a menu for selecting it
SYMB Alt	[Pick one alternative symbol or phrase by pressing a character]
SYMB Del	[delete word]
SYMB Esc	Application Menu (Esc does the same), Front panel button lower left
SYMB Enter	System Menu (Ctrl Esc does the same), Front panel button lower right
SYMB Tab	Switch between active applications (like Shift + Tab)
SYMB Shift	= Shift SYMB; To use upper case of the next characer requiring SYMB

SYMB Ins	<reserved for a charcter>
SYMB Space	<tbd>
SYMB UpArrow	Scroll Up
SYMB DownArrow	Scroll Down
SYMB Home	Scroll Left
SYMB End	Scroll Right
SYMB PageUp	Scroll Up fast
SYMB PageDown	Scroll Down fast

Special Chordons (see Annex 3)

Shift + BackSpace or Shift + Word Left	Delete word from left
Shift + Delete	Delete Word from right
Shift + Word Right	Use CAPS until next Space
Space + Word Right	Space + use CAPS until next Space
Space + 123-ABC	Space + use 123 until next Space
Word Right + 123-ABC	Use 123 until next Space
. ! or ? + Space	Add Space and Shift (...here. **T**hen he...)

5. The GKOS Keyboard Scanning (Polling)

There are several methods to implement the polling of the GKOS keyboard status in order to detect key combinations (chords). Methods A to D below are compatible with the basic typing method where all keys are released after each character and the character appears at the moment keys are released. Typematic delay (autorepeat), however, is not used in Teach Mode.

A. Basic Scanning

In the basic typing and key scanning method, the successively pressed key combinations do not overlap, i.e. all keys must be released before the next character is typed. The typed character is determined by those keys that have been down since the previous condition when there were no keys pressed. The character appears on the screen at the moment all keys are released. Holding down the key combination for e.g. 2 seconds (typamatic delay) will start the automatic repeat of the character. – There is also a more robust variation of this method. Some keys may then remain pressed between characters. The character appears already when the first key is released after depression of any key and it is determined by those keys that have been down since the previous depression of any key. In this case, it is possible to hold down a shift function (G, K, O, S, W or Ü) if it is needed also for the next character (e.g. when typing 'hi' or 'utt'). – Both of these simple standard methods operate perfectly and require very little processing power, allowing typing speeds up to at least 50 wpm.

B. Intelligent Scanning - The recommended way

This method of typing and key scanning is similar to the basic method above except that some amount of overlapping is allowed. The start and stop times of overlapping key presses, as well as the durations of the 'key down' situations are analysed and compared with each other in an intelligent way to decide which key presses go together to form a single character. – This method requires more processing than the basic scanning but is practical with higher typing speeds (60 wpm and beyond). - **A more detailed description is in ANNEX 4.**

C. Teach mode

In teach mode, in order to find the right character, it is possible for the user to change the combination pressed if not all keys are released. While changing the combination, the corresponding character will appear at the (same) cursor position. When all keys are released the character corresponding to the last combination is printed and the cursor is advanced. The typamatic delay (for auto repeat) is not applied in this mode. This function may be applied to symbols including 1 to 3 simultaneous key presses only, showing a blank cursor position for other combinations. - This method makes it possible for a beginner to enter text correctly without looking at the instructions.

D. Special mode

This is an enhancement of Intelligent Scanning so that a different typing technique can be used**:** if the same key (or two) is required in successive characters, supposing they are letters, it is not necessary to release it (or them) in between. The character decision is partly based on the fact that letters are always formed in a certain way, namely 1+0, 0+1, 2+0, 0+2, 2+1 or 1+2 simultaneous keys. Some overlapping is allowed. The standard way of typing also gives a correct result. - The special mode increases the possible typing speed if the analysis of key presses is implemented successfully, but additional practising is required to achieve this (See Annex 3).

E. Game and pointer control

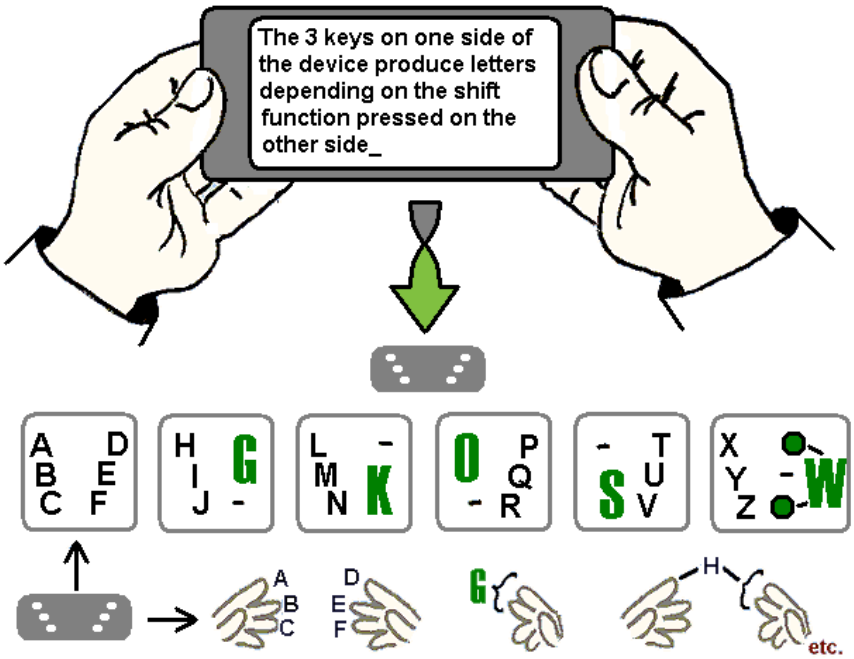
When the GKOS keypad is used for game or pointer control it may function differently and indicate a key press immediately, already at the moment a key is pressed down.

---

6. Document history

Document history		
5 October 2000	The first version published (html)	v1.0.0
	Website version (html)	v2.0.0
	Website version (html)	v2.1.0
10 July 2003	New specification layout (doc)	v3.0.0
4 August 2003	typos in tables corrected (doc)	v3.0.1
8 August 2003	typo corrections and changing inverted ? and ! into ¿ and ¡ in tables	v.3.0.2
20 August 2003	Streamlining of Annex 2	v.3.0.3
25 August 2003	Errors in Tables corrected	v.3.0.3c
26 October 2003	Annex 3 added, Chordon Technique	v.3.0.4
27 October 2003	Annex 3, text improved	v.3.0.5
6 Nov 2003	National characters as word shortcuts	v.3.0.6
10 Nov 2003	Adjustments to Annex 3 parameters	v.3.0.7
2 April 2004	Annex 3 figures updated	v.3.0.9
19 October 2006	Update of Foreword, character tables, special functions and Annex 3; New Annex 4	v.3.1.0
5 Dec 2006	Corrections of typing, and some clarifications. One more picture in Annex 2.	v.3.1.1
7 Dec 2006	Character table adjustments (national and accented characters, and F1...F20). Special Chordons.	v.3.1.2
4 Sep 2008	Characters Ref. 18, 28, 33, 34, 40 and 41 modified	v.3.1.3
4 Mar 2009	Order of characters Ref. 12, 13, 14, 20 and 21 modified	v.3.1.4

ANNEX 1 - How to type on GKOS



Space: press the GKOS '***space bar***' = all 3 keys on the **right** hand side.
Backspace: press the GKOS '***left bar***' = all 3 keys on the **left** hand side.

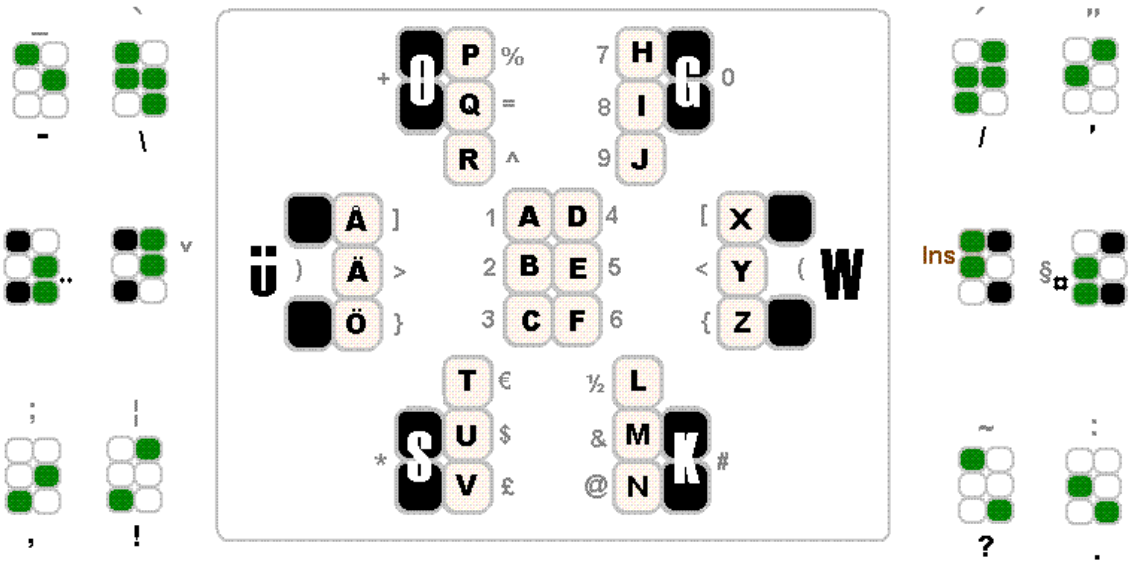
Note that the information above is all you need for typing simple text messages! (In addition, punctuation and case change are of course useful to learn)

The figure shows that **for letters** (and numbers)**, maximum 2 simultaneous key presses are needed per hand** but **whenever one hand presses two keys** (e.g. D+E keys)**, the other hand needs only to press a single key** (e.g. to get "H") **or no key at all** (result is then "G")**.** Letters A to F are just single keys.

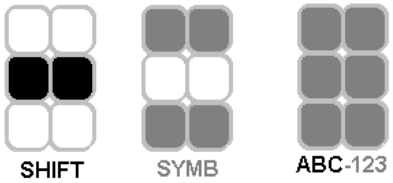
Can you press Ctrl + Alt + Del on a PC keyboard? If you can then you can also enter text, numbers and punctuation on a GKOS keyboard because that combination is similar to the most 'difficult' GKOS letters X, Y, Z.

ANNEX 2 – Character sets and virtual keyboards

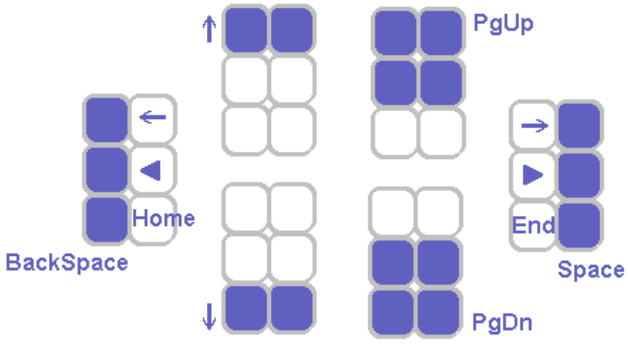
1 The complete character set



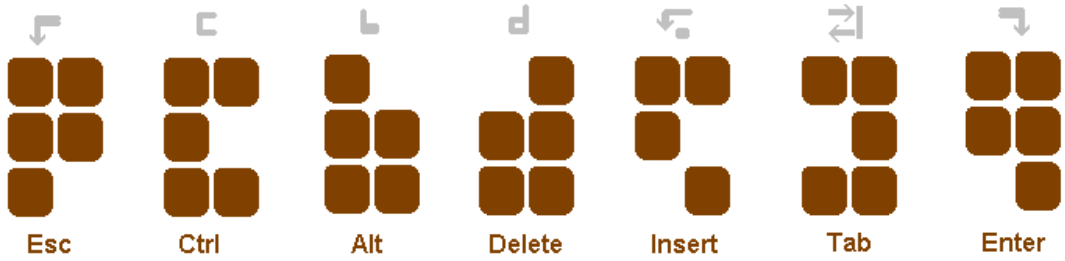
Each black 2-key combination above is a shift function (simultaneous) to obtain the rest of the letters in each group, or just produces the letter marked on it when pressed alone. The main principle is that for frequently used characters, only 1 to 3 simultaneous key presses are needed but for functions more. This way typing is lighter and functions will not be activated by mistake if typing carelessly. The four national characters vary in different national versions of the keyboard.



The **123-ABC** mode change toggles between the two character sets, black and grey, inside the frame shown above. With **SYMB**, single characters can be picked from the parallel set, even outside the frame. For example, *semicolon* is *SYMB* followed by *period*. **SHIFT** is used only for upper case letters. Two consecutive SHIFTs set CAPS LOCK and one SHIFT has an effect only on one character or turns CAPS LOCK off.



Navigation, including space and backspace, is a self evident set. The blue triangles depict *word left/right* (= CTRL + Arrow Left/Right on QWERTY).



Each control function has an appearance that helps to remember its meaning.

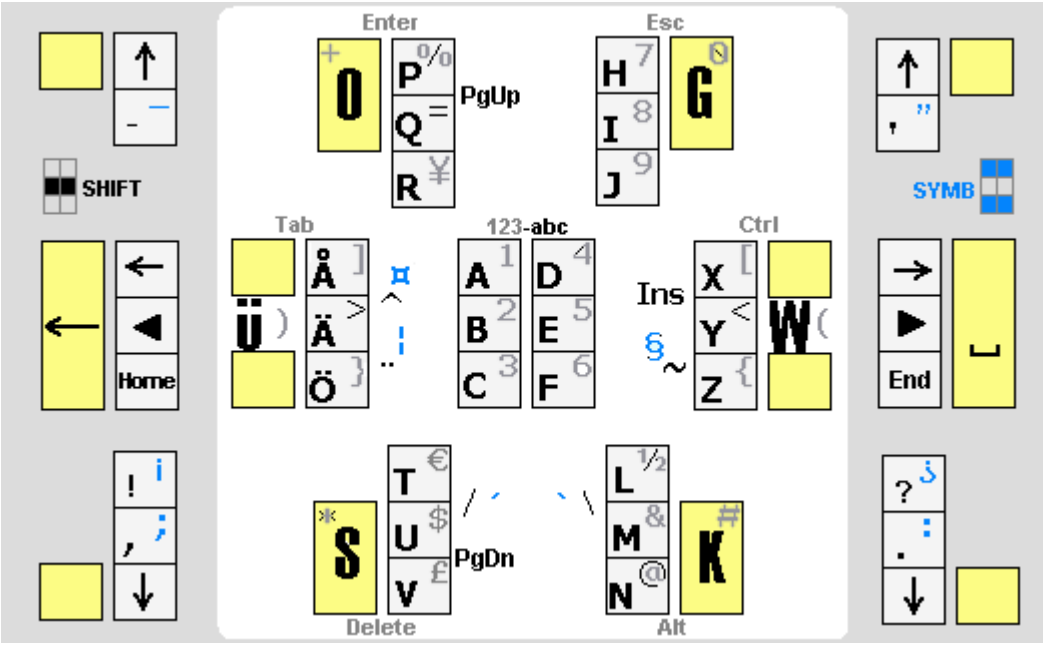
For clarity, the numbers are shown here second time to indicate the position of all number pad symbols.

2 The national character sets

The character locations for letters Ü Ö, Å and Ä above can be different in national versions of the GKOS keyboard layout. They can even be used as shortcuts to common whole words or as shifts to be followed by a letter to get four new sets of 26 characters/words, or even both if the GKOS chordon technique is used: national shift + letter gives a stored word or a special character if typed as a chordon, else it will give directly the word/letter marked on it followed by the next letter(s). See http://gkos.com for more detailed national layout descriptions.

3 Other virtual keyboards

Here is a compact virtual keyboard combining all GKOS characters and functions into a single figure:



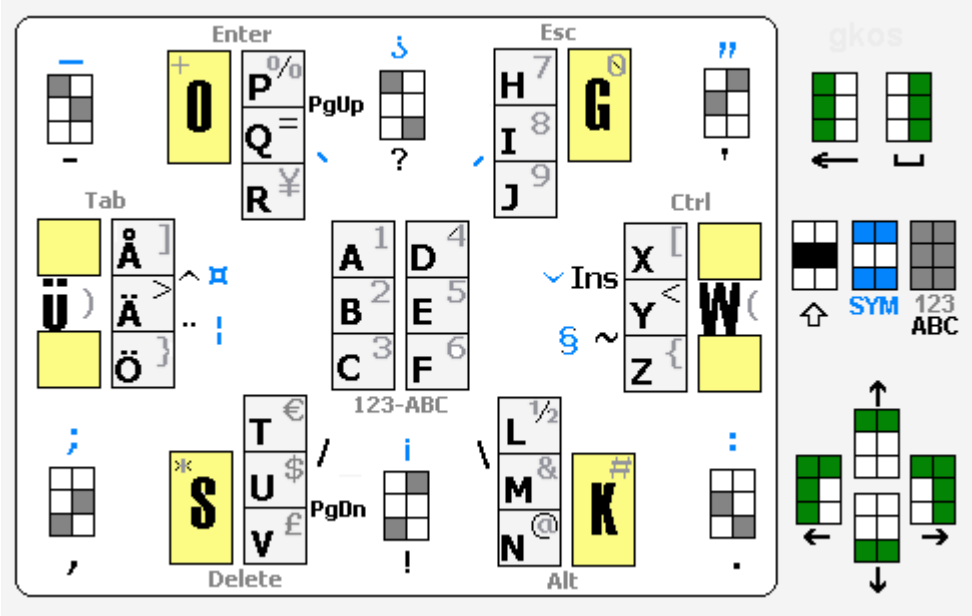
Shift functions are indicated as yellow areas.

Simple virtual keyboards

- letters:

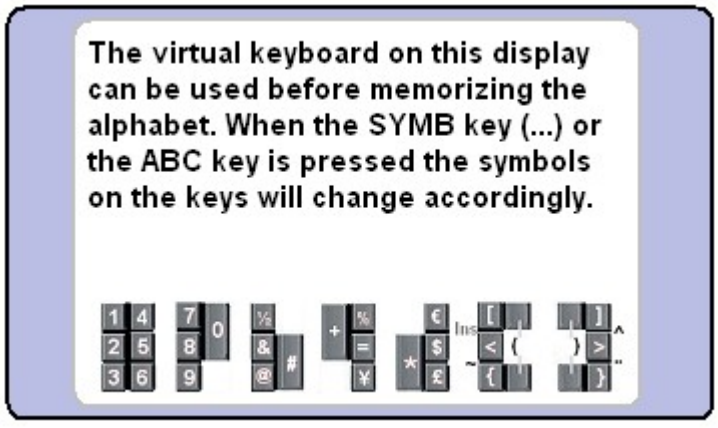
- numbers:

The full set to be printed in credit card size:



Virtual GKOS/QWERTY keyboard:

Keyboard help on the screen:



See http://gkos.com for more virtual GKOS keyboards

ANNEX 3 – Chordon Technique

Entering Chordons on a GKOS Keyboard

Detecting Chord Chains

Typing syllables instead of single letters

It is possible to type chords as a sequence without a complete release of keys inbetween. If there are keys that belong to two (or more) consecutive chords, they are kept depressed also while the chord changes. It is this fact that makes the GKOS chord sequence a ’chordon’ (chord-on-chord, or a chord chain consisting of a series of chords). Between chordons, all keys are released.

Chordon technique is specifically useful with GKOS for at leat three reasons: 1) The 6 nimblest and most similar fingers are used to enter chords, enabling the regular forming of chordons that is necessary; 2) The fact that chords for letters and numbers include only 3 keys maximum makes the detection essentially easier; 3) The 6 fingers never have to leave the GKOS 'home row' which is necessary for being able to type chords in a row without gaps.

Typing a string of letters as a chordon is faster than typing the same letters as separate chords because **typing becomes a parallel process** (next chord can be depressed before the previous one is released). The practical limit of 300 characters/min (**60 WPM**), typical for chording keyboards while using separate chords and fully serial tapping **can then be exceeded by skilful typists**. The typing speed upper limit then only depends on the skill of the user and on the chord detection algorithm.

Typing syllables (typically 2 to 4 letters) as chordons can be learned quite easily after getting otherwise familiar with the GKOS system. First, short common words like ’is’ and ’the’ can be typed as chordons. And then, after being able to type syllables fluently as chordons, even whole longer words can be learned to enter this way. However, the syllables are the key in learning the method and getting the higher text entry speed.

There are letter sequencies that cannot be completely tied together, e.g. ’no’ (C+E+F > A+B), because there are no common keys. Overlapping of chords normally takes place, and must be allowed, if chordon technique is applied by the user. By definition, a chordon may also include chord transitions that just overlap but do not have common keys.

One question still: how to type e.g. ’oo’? Answer 1: a repeat character is used (A+B > A+C) if the national character set can provide it as a national character. Answer 2: If there is no spare GKOS symbol (including 1 to 3 keys) in the national character set, any 2-key punctuation character (.,-’?!) within a chordon can serve as such. As a consequence, real punctuation characters must always be separate chords. Answer 3 (recommended): for a compatible approach, cut the chordon between double letters (go <keys up> od).

The Polling Procedure

The GKOS keyboard polling can be implemented so that both conventional text entry with single separate chords as well as entering of chordons and overlapping chords are all possible at the same time.

How to detect individual chords within a chordon? For every single key release a procedure is started to check back in time to detect the chord just typed and also one or two earlier chords that did not end in a key release.

If we assume that only letters can be combined to form a syllable (combined chords), the detection becomes simpler. As there are never more than 3 keys in a GKOS character (letters and numbers at least), it is required to look back only 2 steps maximum to find the previous key release or an 'all keys up' condition. For example 'bop' is one such rare case where also two previous chords must be detected (B > B+A > B+A+D > at least one key is released next).

The check-back procedure must be clever enough to ignore a key release that is too close after the one that started the procedure because these two (and sometimes even more) key releases may belong to the tail of a single character.

If punctuation characters are not allowed within a chordon (to minimise errors), they can serve as symbols to start **word shortcuts** instead. For example, typing *-a* or *'i* as a chordon can provide a whole word like *answer* or *interesting*.

Parameters

For each chordon, a chord length must be estimated first. The value must be available when the first key release takes place. This parameter will be the basis for the analysis of the whole chord sequence. It should be noted that practically no fixed timing values can be used because typing speeds will vary a lot.

The overall form and structure of the chordon will tell what chords are included, not the absolute lengths of key depressions. In addition to chord length, a guad time must be used during the chord transition. Possible previous chords must also be checked.

A more detailled description of GKOS chord detection is described in ANNEX 4.

ANNEX 4

Seppo Tiainen
Veikkola, FINLAND

Detecting GKOS Chords and Chordons

Two methods in detail

The key combinations, or Chords, entered by the user can either be determined in a simple and robust way, or by monitoring the timing of key depressions. The method based on timing is, however necessary to be able to use the GKOS word shorcuts that are based on Chordons.

Definition of a chordon shortcut word: A Chordon Shortcut is a two-character chordon that is preceded by, and ends in, an all-keys-released condition, and consists of characters of which at least one is a special character belonging to this group of characters (shorcut word tables use the symbols in bold):

, . - ' ? ! Up Arrow **Dn**Arrow **Sh**ift **B**ack**s**pace **Sp**ace

The word shortcuts have been composed according to the principle that the keys pressed are as far as possible the same as (but often fewer than) those when typing the beginning of the word letter by letter. This seems to be a good practice and feels quite natural after having obtained a fluent basic GKOS typing skill. It is even possible to correctly guess the shorcut in many cases.

In case no timing of pressing the keys is monitored, it is enough just to check the combination immediately before the first key is released. It cannot, however, be guaranteed that combinations are detected correctly if there is no requirement to *release all keys between consecutive characters.* This requirement slows down typing. Consequently, Chordons cannot be detected but, on the other hand, the method functions absolutely correct in all situations when typing is done accordingly.

In case timing is monitored, Chordons and chord overlapping can be managed, as well as the case when the chord to detect is just a part of the previous chord. Typing can now be faster but the operation principle becomes a bit fuzzier because fixed time periods cannot be used due to varying typing speeds. A chord time period (duration) must be estimated and applied in the detection of successive combinations. This method can work great and *does everything that the previous simple method does plus much more*, but it can be spoiled easily with bad design.

Note that the key scanning routines must have the highest prioroty of all processes to have a proper response. The descriptions below are intended to assist in implementing the key scanning procedures of a prorotype or a final product.

GKOS chord detection without timing

There are two triggers for this procedure: Key up and Key down, indicating the transition of the state of any key. One boolean variable (R, Ready to read) is necessary:

```
Key down =>        R = TRUE; return;

Key up =>          In case R == FALSE: Return;
                   In case R == TRUE: R = FALSE;
                              Read Chord value immediately before the
                              the key went up and output the value;
                              Return;
```

GKOS chord detection with timing

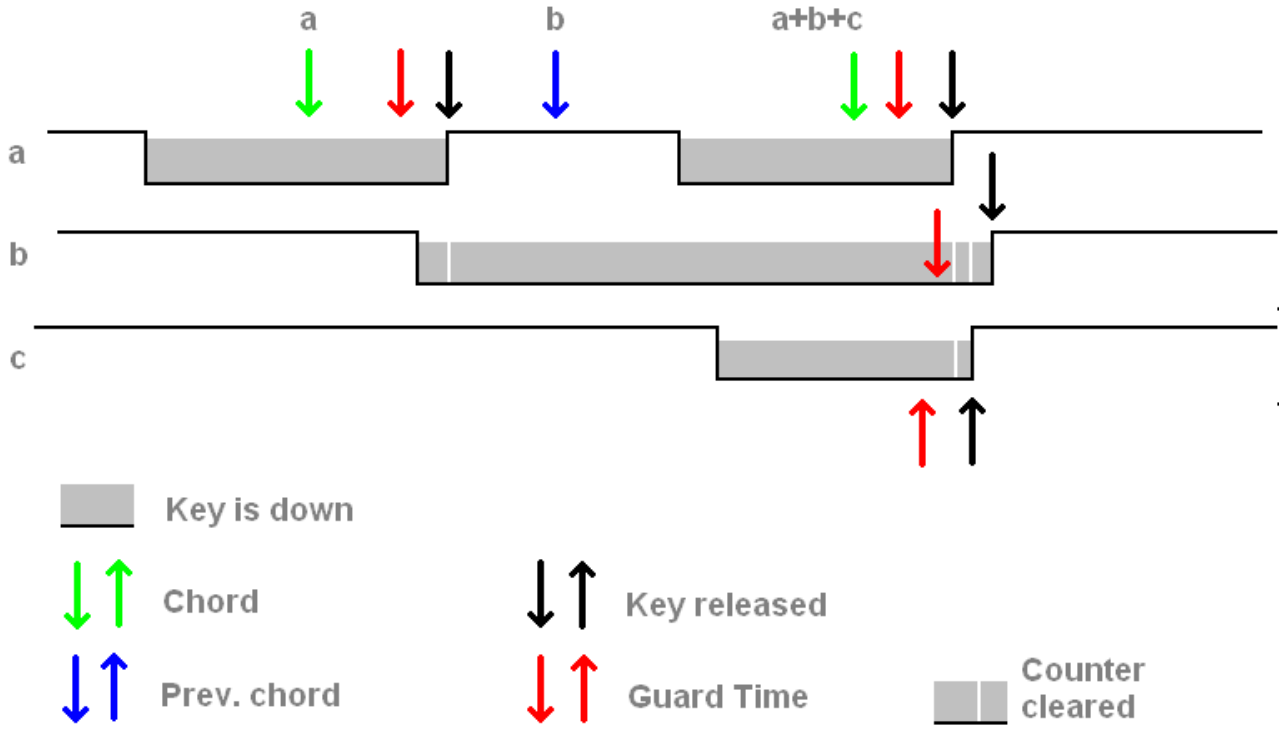
There is a **Key Timer** (up counter) for each of the 6 keys. They are incremented every 10 ms while the corresponding key is held down. Maximum counter value is 255 = 2,55 s. Estimated **Chord Period Tc** and **Guard Time Tg** are also needed. Initial values for these variables are: Tg = 80 ms (= 8 counts), Tc = 160 ms (= 16 counts). The procedure starts when any key is released:

```
Key up =>          Get the lowest value of all Key Timers exceeding Tg and give
                              Tc that new value;

                   Get the chord value (1...63) of those keys that have Key
                              Timer value greater than 0.5 x Tc;  // => Present Chord

                   If the chord value obtained is zero (none of the Key Timers
                              exceeds Tg), just get the chord value of all non-zero
                              Key Timers. Do not repeat this check within Tg.
                                        // fast hit; => the Present Chord

                   Get the chord value of those keys that have Key Timer
                              value greater than 1,75 x Tc;  // => Previous Chord

                   Clear all Key Timers;
                              Set Tc = 160 ms (= 16 counts, default);

                   Output the Previous Chord (if any) and the Present Chord;

                   Update Guard Time value:
                              Tg = 0,2 x Tc (keep within a range of 60...200 ms);
                              // Optionally Tg can be kept constant: Tg = 80 ms

                   If this is a transition to All Keys Up condition, indicate it
                                        (for outputting the <AKU> signal).

                   Return;
```

It shall also be monitored whether the **Autorepeat Timer** exceeds **Ta**. Parameter Ta has a fixed preset value of 0,5 s to 1,0 s (50...100 counts). The timer is increased every 10 ms and cleared each time the staus of the keyboard changes. It the timer reaches this Ta ('typamatic delay'), the non-zero Chord value at that moment will be ouput repeatedly at 10 ms intervals as long as there is no change in the keyboard status. If there is a change, the the repeating is stopped, the timer cleared, and no

Chord values are sent until all keys have been released and a new Chord entered. Some Chord values, however, are not repeated (Shift, Ctrl etc).

Here is an example of timing-based detection when pressing key a, then key b, then three keys a, b and c together:



In the example above, the three characters (a, b and Backspace) are entered as a Chordon, which meas that there is no simultanous release of *all* keys between the characters.

Seppo Tiainen
Veikkola, FINLAND