Addressing Ten Fingers
*** a tool as a substitute for human voice ***

Manaka Shunyuu

Abstract：
Processing combined Japanese Kana characters which are made by PC's Japanese 106 keyboard through a compact speech soft and so on in real time, we emit a sound over a speaker. If we specify speaking speed and pitch pointing a rectangle zone in a display with a head pointer and we send the two items of information to a compact speech soft or an outer interface and sound source, we get a singing sound by artificial voice.

1. Key arrangement of 50 "ON"
The consonants of Kana character correspond like the following:

・な line、た line、さ line、か line、あ line ⟺ A、S、D、F、C(C or V)
・わ line、ら line、や line、ま line、は line ⟺ カタカナ・ひらがな・ローマ字+A、S、D、F、C(C or V)
・わ line：わ、を、ん、vacant、vacant

C or V needs a modification of the configuration file. カタカナ・ひらがな・ローマ字 is a kind of Shift key, however, because it only raises a flag, it is not necessarily needed that we press it before A、S、D、F、C(C or V).
The vowels of Kana character correspond like the following:

・a、i、u、e、o ⟺ '.'('.' or ',')、L、;、:、]

'.' or ',' needs a modification of the configuration file.
The above is lower type. Upper type is the following:

・な line、た line、さ line、か line、あ line ⟺ Q、W、E、R、F
・わ line、ら line、や line、ま line、は line ⟺ カタカナ・ひらがな・ローマ字+Q、W、E、R、F
・a、i、u、e、o ⟺ L、O、P、@、[

The Enter function is given to vowel key. For example, on lower type

・き：F↓、L↓

2. Voiced sound, p-sound, small character, long sound
The voiced sound and p-sound of Kana character correspond like the following:

・p-sound：O
・voiced sound：P

The small characters of や line and あ line correspond like the following:

・small character of や line：@
・small character of あ line：[

The small character "っ" corresponds like the following:

・small character "っ"：無変換

The long sound "ー" corresponds like the following:

・long sound "ー"：Space

Upper type is the following:

・p-sound：9
・voiced sound：0
・small character of や line：−
・small character of あ line：∧

The nearest key line is common to lower type and upper type. Ctrl and Alt are cancellation key.

・cancellation key：Ctrl、Alt

3. Kana combination

Generally, considering the working efficiency, we make a bundle of some Kana characters in Kana characters which are sent to a speech soft or an outer interface. For example, on lower type

・きっ：F↓、無変換↓、L↓
・きん：F↓、変換↓、L↓
・きゅっ：F↓、@↓、無変換↓、L↓、;↓
・きゅん：F↓、@↓、変換↓、L↓、;↓
・きゅーっ：F↓、@↓、無変換↓、Space↓、L↓、;↓
・きゅーん：F↓、@↓、変換↓、Space↓、L↓、;↓
・びん：カタカナ・ひらがな・ローマ字↓、C↓、P↓、変換↓、L↓
・ぴん：カタカナ・ひらがな・ローマ字↓、C↓、O↓、変換↓、L↓

We use not 50 "ON" but 変換 in "ん" here. If it is "きゅ...", we press the vowel key L of "き" and the vowel key ； of "ゅ" in this order last. If want to enlarge the input number of Kana characters per a unit of time, we need to press keys required in a short time using the ten fingers before pressing vowel keys.

4. Voice output

We adopt espeak-ng as speech soft. The command line style is the following:

```
espeak-ng -v ja+f5 -a 175 -s 126 -p 50 き &
```

We fix -a. If we use it for conversation, speaking speed -s and pitch -p are fixed.

If we use it for singing, we must change them in real time. If a helper is, speaking speed -s is given it putting cursor on a rectangle zone of a display. If a helper is not, setting speaking speed -s a little small, we overlap two espeak-ngs. If the overlap is, getting PID of the previous espeak-ng with ps -t, we kill it.

ps t > ttt
kill -9 PID

Pitch -p is given putting cursor on another kind of a rectangle zone of a display. The place is controlled with a beam from a head pointer fixed on the user's head in Figure 1. However, If a helper is, the number of cursors becomes two.

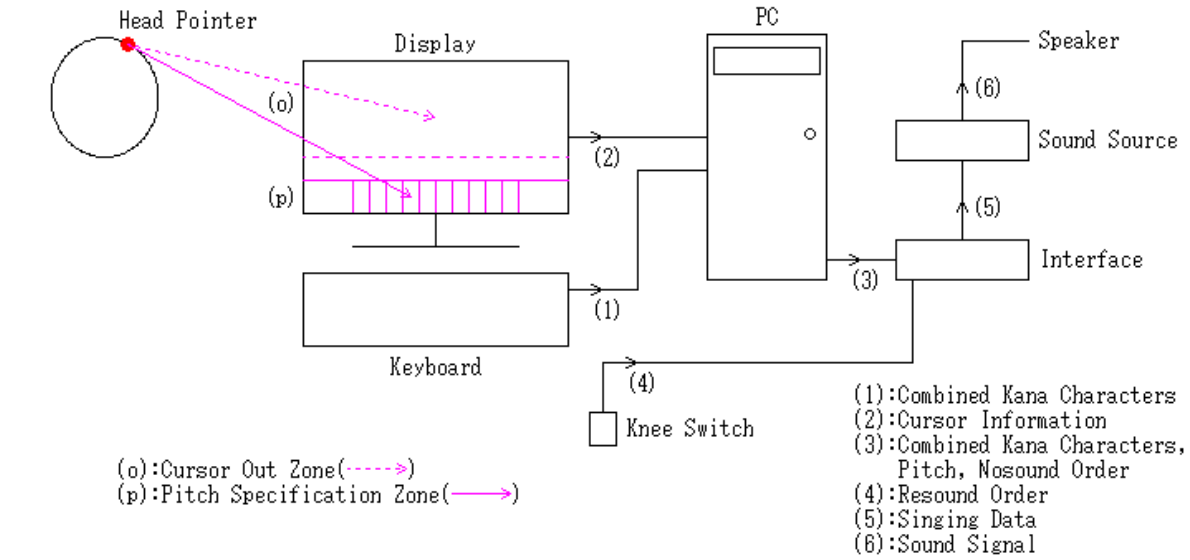

Figure 1

espeak-ng has software sound source. We can emit a sound of combined Kana characters not only by using it but also by playing a voice file with a command line music player.

```
mpg321 -q きっ.mp3 &
ogg123 -q きっ.ogg &
```

In this case too, just like the case of espeak-ng , setting playing time a little long, we use kill command if two music players overlap. The technique is effective in the case that we want to reflect user's voice quality on artifitial voice.

We must prepare voice files by the number which corresponds to 50 "ON" and pitches of Kana combinations. The form is as follows:

・[i][i_][j][j_][k][l][m][n].mp3
・i：consonant(1〜5 ⟺ あ line〜な line、6〜10 ⟺ は line〜わ line)
・i_：vowel(1〜5 ⟺ a〜o)
・j：consonant of small character(あ line、や line)(1、2 ⟺ あ line、や line)
・j_：vowel of small character(あ line、や line)(1〜5 ⟺ a〜o)
・k：small character "っ"(1)、ん (2)
・l：long sound "ー"(1)
・m：voiced sound(1)、p-sound(2)
・n：pitch(1+0,1+1,1+2, ...)

For example

・びゅーん：[6][2][2][3][2][1][1][5].mp3

5. Outer sound source

When we use outer sound source, general MIDI machines may not be able to correspond to the sound emission of Kana characters, because the input of them is signal which originates in keyboard and guitar and they will not cover character information. On internet investigation, the elements of the system will become like the following:

・Japanese 106 keyboard(information of combined Kana characters)
・display and head pointer(information of cursor)
・interface(Bus connection or USB connection ; equivalent to Arduino)
・sound source(singing composition engine ; equivalent to VOCALOID-board, eVY1 shield)

The information of combined Kana characters and the information which origines in a place of cursor are sent from PC to interface. When cursor is in a rectangle zone in the rectangular zone which is drawn in the display in Figure 1, if a vowel key of

the PC's keyboard is pressed, combined Kana characters is emitted at the corresponding pitch and if cursor moves to the outside of the zone during the sound emission, the sound emission is terminated. We must put cursor back in the zone in order to reemit a sound. When cursor is in the zone, if we press knee switch, if combined Kana characters has been emitted once, we emit the memorized combined Kana characters getting a pitch even if there is no press of a vowel key of PC's keyboard.

・press of a vowel key of PC's keyboard : note off and note on(infinite long sound)
・press of knee switch(no renewal of combined Kana characters) : note off and note on(infinite long sound)
・movement of cursor to the outside of the zone : note off

note off means termination of the sound emission of combined Kana characters.

If we adopt the composition like Figure 1, the function prototype of the functions which correspond to the above actions will become like the following:

```
_add2_kana(char *kana, int pitch)
_add2_resound(int pitch)
_add2_nosound(void)
```

On `_add2_resound`, we assume that we can detect press of knee switch.

If we adopt outer sound source, we can emit an infinite long sound. When we adopt speech soft or voice file, we realize it using an extender or an elementary program. When we only use an extender, it is not necessarily needed that we stick to MIDI standard. It is arranged between PC and Speaker.

・PC $\Rightarrow$ Extender $\Longrightarrow$ Speaker
・$\Rightarrow$: finite long sound、$\Longrightarrow$: infinite long sound

The composition of the signals between PC and Extender becomes like the following:

・(3′) : control signal、data signal、voice signal

Extender does a processing(addition of long sound) if the voice signal of input enters long sound part. If we use Extender, we can reduce the lag before the first processing on speech soft or music player of combined Kana characters.

An elementary program only can realize simple long sound. First, we make a copy of music player.

・cp -a mpg321 mpg3cp

We prepare vowel long sound files which are used for addition.

・ah.mp3、ih.mp3、uh.mp3、eh.mp3、oh.mp3

The following is the examples.

・び(ー) $\Longrightarrow$ ih.mp3
・びゅ (ー) $\Longrightarrow$ uh.mp3
・びゅ : `[6][2][2][3][0][0][1][5].mp3`

We process combined Kana characters with speech soft or music player and we process vowel long sound with music player.

```
espeak-ng -v ja+f5 -a 175 -s 12 -p 50 びゅ &
mpg321 -q [6][2][2][3][0][0][1][5].mp3 &

mpg321 -q uh.mp3 &
mpg3cp -q uh.mp3 &
```

We assume that a process which is generated with `&` on combined Kana characters is $\text{process}_{\text{pb1}}$, the kill of it is kill_1, a process of mpg321 and a process of mpg3cp which are generated with `&` on a vowel long sound file which is used for addition are

$process_{pb321}$ and $process_{pb3cp}$, the kills of them are $kill_{321}$ and $kill_{3cp}$ and the delay times are Δ_{d1} and Δ_{d}. The event order becomes like the following:

・$process_{pb1}$、""、Δ_{d1}、$process_{pb321}$、$kill_{1}$、Δ_{d}、
$process_{pb3cp}$、$kill_{321}$、Δ_{d}、
$process_{pb321}$、$kill_{3cp}$、Δ_{d}、
$process_{pb3cp}$、$kill_{321}$、Δ_{d}、
$process_{pb321}$、$kill_{3cp}$、Δ_{d}、
.....
.....

In Δ_{d1} and Δ_{d}, we use XPending before XNextEvent.

```
if(XPending(...)){
XNextEvent(...);
.................
}
```

When we emit a next combined Kana characters, we must kill processes which have the possibility of continuation beforehand.

$kill_{1}$、$kill_{321}$、$kill_{3cp}$

In order to putting it into practice, it is required that a vowel long sound is flat and a long sound after Δ_{d1} of combined Kana characters and it are in the relation of being flat.

6. English

There is a possibility that we can appropriate Japanese Kana characters to English as mere phonetic signs. We will not be able to cover all the phonetic signs which a native uses regularly, however, the meaning of a sentence will be understood from the context. If there are some phonetic signs that just these are required absolutely, we corespond to them with the use of unused or vacant 50 "ON", the addition of Kana line key(left finger), for example, G(T) or the diversion of right Alt which is cancellation key to the second カタカナ・ひらがな・ローマ字. In hand sign, we correspond to them with the upper shift or lower shift of left hand. We emit them using speech soft, voice file or outer sound source.

7. Language education of animal

If we add some devices to the key arrangement of Japanese 106 keyboard, we may be able to make language education to animal. The new keyboard has the following spec:

・な line、た line、さ line、か line、あ line $\Longleftrightarrow$ A、S、D、F、G
・わ line、ら line、や line、ま line、は line $\Longleftrightarrow$ Z、X、C、V、B
・a、i、u、e、o $\Longleftrightarrow$ K、L、;、:、]
・p-sound：O
・voiced sound：P
・small character of や line：@
・small character of あ line：[
・small character "っ"：無変換
・long sound "ー"：Space(the same size as 無変換)
・ん：変換
・cancellation key：Ctrl(moved forward)

The number of keys are 24.

・We enlarge the area of a key top and fix a snack food namely an oyatsu to it. We make a device that the oyatsu is set free after key press to the key.
・The number of keyboards is larger than 1.

At first, on a simple Kana character, we give a keyboard which an oyatsu is placed on a consonant key of, next we give a keyboard which an oyatsu is placed on a vowel key of and we emit the Kana character if the vowel key is pressed. Finally, we give one keyboard which an oyatsu is placed on all keys required of. However, we make a device that on Kana line key, if unrelated keys are pressed after pressing of keys required, on keys required except Kana line key and vowel key, if unrelated keys are pressed, on vowel key, if not all the other keys required are pressed and when plural vowel keys must be pressed, if the press order is wrong, the oyatsu is not got. After that, we decrease the number of oyatsues gradually following the following steps:

(1)No oyatsues of keys required except vowel keys are. We turn on their LED lights.
(2)No oyatsues of vowel keys are. We also turn on their LED lights. Making an animal hear the Kana unit, we give an oyatsu if it is emitted.
(3)No LED lights are. Making the animal hear the Kana unit, we give an oyatsu if it is emitted.
(4)Showing an oyatsu, we make the animal hear the name "おやつ". We give an oyatsu if "おやつ" is emitted.

The order of categories of the words which we teach is as follows:

(1)Kana characters
(2)physical existence, state, event of the outside world
(3)number, order, time, age, sex, full name
(4)desire, intention, assertion
(5)impression, evaluation, guess

For example, on (2) and (3), we teach the following words:

・おやつ、わたし、あなた、あつい、さむい、ひとつ、ふたつ、さいご、あした

An animal who learns words must satisfy the following conditions:

・The intelligence and memory ability are high
・It has desire for knowledge and intellectual curiosity
・It has self-control
・quiet and calm
・female

Besides, on an animal who lives in groups, we must choose one of the following two situations:

・We isolate it from a group
・We do not isolate it from a group

On which animal to make language education to, whether we isolate it from a group or not and so on, it will be wise of us to judge them after we gather data on animals making simulation with AI animal.

If an anmal has made progress in language as far as it exchanges a daily conversation with a human using artifitial voice, we teach hand sign. However, the animal must have both arms and ten fingers, and besides be able to move them skillfully.

8. Program

The program is executable in linux and BSD if Xlib is.

・Slackware15、jammypup64_9.8、netBSD10RC4(no sound emission)

If we adopt the composition of Figure 1, we will be able to use hardware on the market as it is. It seems that the software for MIDI keyboard in USB connection is rather in circulation. The main modification parts are the information source of pitch, note on, note off and the combination of note on, note off.

If `espk_af` is 0, espeak-ng is adopted and if `espk_af` is 1, a music player is adopted.

```
#define espk_af 0/*1*/
#define AUDIO "espeak-ng"
```

```
#define Audio "mpg321"
#define X_PATH "/root/gcc/v_mp3"
#define FT "mp3"
```

・ja_dict：47652 bytes

Even if `espk_af` is 0, we use a music player for "ん". `Audio, X_PATH, FT` are the name, the directory of the voice file, the extension.

```
#define espk_af /*0*/1
#define AUDIO "mpg321"
#define X_PATH "/root/gcc/v_mp3"
#define FT "mp3"
```

・びゅーん：`[6][2][2][3][2][1][1][5].mp3`

In the above form, the pitch 5 corresponds to 4 in the p rectangular zone in the display.

If we want not to monitor combined Kana characters and pitch, we adopt the following:

```
#define kanaDATA 0
```

If the leading character of the name of the executable file is @, the key assignment is upper type.

・upper type：@add10
・lower type：add10

We place the configuration file blecfg in /root. If we adopt C or V in lower type, in the configuration file blecfg, we set the item like the following:

```
no.17::2    /* lt1 */
```

If we adopt '.' or ',' in lower type, we set the item like the following:

```
no.18::2    /* rt1 */
```

If we do not use kill command, we set the item like the following:

```
no.26::0    /* break */
```

The Japanese keys are 無変換, 変換, カタカナ・ひらがな・ローマ字. If the two keys among them are pressed at the same time, a possibility of disorder may occur. Therefore, if the number of keys required except vowel key is large, there is a case that we must divide them into two parts. The division of presses can be avoided if we swap wirings of Japanese 106 keyboard physically.

・swap：left Alt ⟺ 無変換
・swap：right Alt ⟺ 変換
・swap：right Ctrl ⟺ カタカナ・ひらがな・ローマ字

If swapped, we set the item like the following:

```
no.25::1    /* ideal */
```

Because cancellation keys are also swapped if the above work is done, the geometrical positions are invariable.
If we make physical swaps of wirings and modifiy the configuration file blecfg, the key assignment from 無変換 to カタカナ・ひらがな・ローマ字 changes partly like the following:

・before：＂っ＂、＂ー＂、＂ん＂、shift
・after　：＂っ＂、＂ん＂、＂ー＂、shift

****

Addressing Ten Fingers
*** 声帯による声の代用としてのツール ***

間中春由

アブストラクト：
PC の日本語 106 キーボードで作成された日本語カナ文字の結合語をコンパクトなスピーチソフトなどでリアルタイムに処理してこれをスピーカーから放音します。ディスプレイ内の矩形ゾーンをヘッドポインターでポイントして話速とピッチを指定し、これらの情報をコンパクトなスピーチソフトまたは外部のインターフェイス・音源に送ると人工音声による歌唱が実現されます。

1. 50 音のキー配置
カナ文字の子音は以下のように対応させます。

・な行、た行、さ行、か行、あ行 ⟺ A、S、D、F、C(C or V)
・わ行、ら行、や行、ま行、は行 ⟺ カタカナ・ひらがな・ローマ字+A、S、D、F、C(C or V)
・わ行：わ、を、ん、vacant、vacant

C or V は設定ファイルの修正が必要です。カタカナ・ひらがな・ローマ字は一種の Shift キーですが、フラグを立てるだけなので必ずしも A、S、D、F、C(C or V) の前にプレスする必要はありません。
カナ文字の母音は以下のように対応させます。

・a、i、u、e、o ⟺ ’.’(’.’ or ’,’)、L、; 、: 、]

’.’ or ’,’ は設定ファイルの修正が必要です。
以上は lower タイプであり、upper タイプは以下のようになります。

・な行、た行、さ行、か行、あ行 ⟺ Q、W、E、R、F
・わ行、ら行、や行、ま行、は行 ⟺ カタカナ・ひらがな・ローマ字+Q、W、E、R、F
・a、i、u、e、o ⟺ L、O、P、@、[

Enter 機能は母音キーにあります。例えば、lower タイプでは

・き：F↓、L↓

2. 濁音・半濁音・小文字・長音
濁音・半濁音は以下のように対応させます。

・半濁音：O
・濁音：P

や行とあ行の小文字は以下のように対応させます。

・や行の小文字：@
・あ行の小文字：[

小文字"っ"は以下のように対応させます。

・小文字"っ"：無変換

長音"ー"は以下のように対応させます。

・長音"ー"：Space

upper タイプでは以下のようになります。

・半濁音：9
・濁音：0
・や行の小文字：–
・あ行の小文字：∧

手前のキー行は lower タイプ、upper タイプに共通です。Ctrl、Alt はキャンセルキーです。

・キャンセルキー：Ctrl、Alt

3. カナ結合

スピーチソフトまたは外部のインターフェイスに送るカナ文字は、一般には作声効率を考慮していくつかをひとまとめにします。例えば、lower タイプでは

・きっ：F↓、無変換↓、L↓
・きん：F↓、変換↓、L↓
・きゅっ：F↓、@↓、無変換↓、L↓、;↓
・きゅん：F↓、@↓、変換↓、L↓、;↓
・きゅーっ：F↓、@↓、無変換↓、Space↓、L↓、;↓
・きゅーん：F↓、@↓、変換↓、Space↓、L↓、;↓
・びん：カタカナ・ひらがな・ローマ字↓、C↓、P↓、変換↓、L↓
・ぴん：カタカナ・ひらがな・ローマ字↓、C↓、O↓、変換↓、L↓

ここでは"ん"は 50 音ではなく変換を利用します。"きゅ..."においては、最後に"き"の母音キー L、"ゅ"の母音キー ; をこの順でプレスします。単位時間当たりのカナ文字入力数を大きくするためには、母音キーをプレスする前に両方の指を用いて短時間で所要キーをプレスする必要があります。

4. 音声出力

スピーチソフトとしては espeak-ng を採用します。espeak-ng のコマンドラインは以下のようなスタイルになっています。

```
espeak-ng -v ja+f5 -a 175 -s 126 -p 50 き &
```

-a は固定します。会話に利用する場合は話速-s とピッチ-p は固定します。

歌唱に利用する場合はこれらをリアルタイムに変えなければなりません。ヘルパーがいる場合は、ヘルパーがディスプレイの矩形ゾーンにカーソルを置くことによって話速-s を与えます。ヘルパーがいない場合は、話速-s を小さめに設定して espeak-ng をオーバーラップさせます。このオーバーラップにおいては、前の espeak-ng の PID を ps t で取得し、これを kill します。

ps t > ttt
kill -9 PID

ピッチ-p はディスプレイの別の矩形ゾーンにカーソルを置くことによって与えられます。このカーソルの位置は図 1 のユーザーの頭部に取り付けたヘッドポインターからのビームで制御します。ただし、ヘルパーがいる場合はカーソルが二つ必要になります。

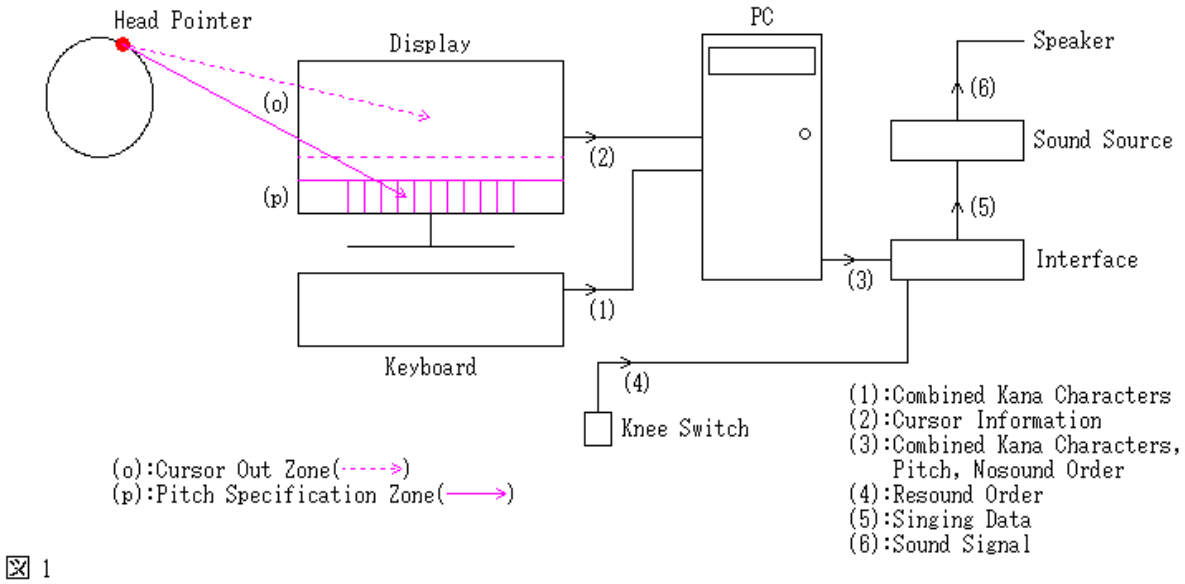

図 1

espeak-ng はソフトウェア音源を有しています。ソフトウェア音源ではなく音声ファイルを音楽プレイヤーで処理することによってもカナ文字の結合語を放音することができます。

```
mpg321 -q きっ.mp3 &
ogg123 -q きっ.ogg &
```

この場合も espeak-ng の場合と同様、処理時間を長めにし、オーバーラップする場合は kill 処理を行います。この手法はユーザーの声質を人工音声に反映させたい場合に有効です。

音声ファイルは 50 音とカナ結合のピッチに対応する分だけ用意しなければなりません。以下は音声ファイルの書式です。

・[i][i_][j][j_][k][l][m][n].mp3
・i：子音 (1〜5 ⟺ あ行〜な行、6〜10 ⟺ は行〜わ行)
・i_：母音 (1〜5 ⟺ a〜o)
・j：小文字 (あ行、や行) の子音 (1、2 ⟺ あ行、や行)
・j_：小文字 (あ行、や行) の母音 (1〜5 ⟺ a〜o)
・k：小文字"っ"(1)、ん (2)
・l：長音"ー"(1)
・m：濁音 (1)、半濁音 (2)
・n：ピッチ (1+0,1+1,1+2, ...)

例えば

・びゅーん：[6][2][2][3][2][1][1][5].mp3

5. 外部音源

外部音源を用いようとすると一般の MIDI 機器では対応できないかもしれません。一般の MIDI 音源の入力は鍵盤やギターに由来する信号であり、文字情報はカバーしていないでしょうから。文献を調べたところ図 1 のような構成になりそうです。

・日本語 106 キーボード (結合カナ文字の情報)
・ディスプレイ、ヘッドポインター (カーソルの情報)
・インターフェイス (Bus 接続または USB 接続；Arduino 相当)
・音源 (歌唱合成エンジン；VOCALOID-board、eVY1 shield 相当)

PC からインターフェイスには結合カナ文字の情報とカーソルの位置に由来する情報が送られます。図 1 のディスプレイに描かれた矩形列のある矩形にカーソルがある場合に PC のキーボードの母音キーをプレスすると、対応するピッチで結合カナ文字が放音され、もし放音中にカーソルが矩形列の外に移動すれば放音をやめます。再び放音するためにはカーソルを矩形列内に戻さなければなりません。カーソルが矩形列内にある場合にニースイッチをプレスすると、一度でも結合カナ文字が放音された後では PC の

キーボードの母音キーをプレスしなくても、記憶されている結合カナ文字をピッチを取得して放音します。

・PC のキーボードの母音キーのプレス：note off と note on(無限長音)
・ニースイッチのプレス (結合カナ文字の更新なし)：note off と note on(無限長音)
・カーソルの矩形列の外への移動：note off

note off は結合カナ文字の放音の終了を意味します。
　図 1 のような構成にした場合、上記のアクションに対応する関数の関数プロトタイプは以下のようになるでしょう。

```
_add2_kana(char *kana, int pitch)
_add2_resound(int pitch)
_add2_nosound(void)
```

`_add2_resound` に関してはニースイッチのプレスを検知できると仮定しています。
　外部音源では無限長音が可能です。スピーチソフトや音声ファイルを用いる場合はエクステンダーまたは初等的なプログラムで無限長音を実現します。エクステンダーだけを用いる場合は必ずしも MIDI 規格に拠る必要はありません。エクステンダーは PC と Speaker の間に配置します。

・PC $\Rightarrow$ Extender $\Longrightarrow$ Speaker
・$\Rightarrow$: finite long sound、$\Longrightarrow$: infinite long sound

PC とエクステンダーの間の信号の構成は以下のようになります。

・(3′) : 制御信号、データ信号、音声信号

エクステンダーは入力の音声信号が長音パートに入ったらプロセッシング (長音の継ぎ足し) を行います。エクステンダーを用いれば結合カナ文字のスピーチソフトや音楽プレーヤーに関する第一処理の前のラグを短くすることができます。
　初等的なプログラムが実現できるのは簡易無限長音です。まず音楽プレイヤーのコピーを作ります。

・cp -a mpg321 mpg3cp

継ぎ足す母音長音ファイルを用意します。

・ah.mp3、ih.mp3、uh.mp3、eh.mp3、oh.mp3

以下はその例です。

・び (ー) $\Longrightarrow$ ih.mp3
・びゅ (ー) $\Longrightarrow$ uh.mp3
・びゅ : `[6][2][2][3][0][0][1][5].mp3`

結合カナ文字はスピーチソフトまたは音楽プレーヤーで処理し、母音長音ファイルは音楽プレーヤーで処理します。

```
espeak-ng -v ja+f5 -a 175 -s 12 -p 50 びゅ &
mpg321 -q [6][2][2][3][0][0][1][5].mp3 &
```

```
mpg321 -q uh.mp3 &
mpg3cp -q uh.mp3 &
```

結合カナ文字に関する&付きで生成されたプロセスを $\text{process}_{\text{pb1}}$、その kill を kill_1、継ぎ足す母音長音ファイルに関する&付きで生成された mpg321、mpg3cp のプロセスを $\text{process}_{\text{pb321}}$、$\text{process}_{\text{pb3cp}}$、それらの kill を kill_{321}、$\text{kill}_{3\text{cp}}$、遅延時間を Δ_{d1}、Δ_{d} とします。イベントオーダーは以下のようになります。

・$\text{process}_{\text{pb1}}$、""、$\Delta_{\text{d1}}$、$\text{process}_{\text{pb321}}$、$\text{kill}_1$、$\Delta_{\text{d}}$、

$process_{pb3cp}$、$kill_{321}$、Δ_d、
$process_{pb321}$、$kill_{3cp}$、Δ_d、
$process_{pb3cp}$、$kill_{321}$、Δ_d、
$process_{pb321}$、$kill_{3cp}$、Δ_d、
.....
.....

Δ_{d1}、Δ_d においては XNextEvent の前に XPending を用います。

```
if(XPending(...)){
XNextEvent(...);
.................
}
```

次の結合カナ文字の放音を行う場合は、その前に存続している可能性のあるプロセスを kill しなければなりません。

$kill_1$、$kill_{321}$、$kill_{3cp}$

実用に供するためには母音長音ファイルがフラットで、かつ結合カナ文字の Δ_{d1} 経過時の長音がこれとフラットな関係にあることが求められます。

6. 英語

日本語カナ文字を単なる発音記号として英語に流用することが考えられます。ネイティブが常用する発音記号の全てを網羅することはできないでしょうが、前後の単語の意味から文意は通じると思います。これだけは絶対必要だとされる発音記号がもしあれば、使用しないまたは開いている 50 音の使用、またはカナ行キー (left finger) の追加、例えば G(T)、またはキャンセルキー右 Alt の第二カタカナ・ひらがな・ローマ字への転用によって対応します。手話では左手の上方または下方へのシフトによって対応します。これらはスピーチソフトまたは音声ファイル、外部音源を用いて放音します。

7. 動物の言語教育

キーボードを工夫すれば動物の言語教育を行うことができるかもしれません。新キーボードは以下のようなスペックとします。

・な行、た行、さ行、か行、あ行 $\Longleftrightarrow$ A、S、D、F、G
・わ行、ら行、や行、ま行、は行 $\Longleftrightarrow$ Z、X、C、V、B
・a、i、u、e、o $\Longleftrightarrow$ K、L、;、:、]
・半濁音：O
・濁音：P
・や行の小文字：@
・あ行の小文字：[
・小文字"っ"：無変換
・長音"ー"：Space(無変換と同じ大きさ)
・ん：変換
・キャンセルキー：Ctrl(前方に移動させる)

キーは以上の 24 個だけとします。

・キートップの面積を大きくし、おやつを固定する。プレス後におやつが解放される工夫を施す。
・キーボードは複数用意する

初めは単純なカナ文字に関して、子音キーにおやつを配置したキーボード、母音キーにおやつを配置したキーボードの順に与え、母音キーをプレスしたらカナ文字を放音します。最終的には所要キー全てにおやつを配置したキーボード一つを与えます。ただし、カナ行キーに関しては所要キーのプレス後に無関係なキーをプレスしたら、カナ行キー・母音キー以外の所要キーに関しては無関係なキーをプレスしたら、母音キーに関しては他の所要キーを全てプレスしないと、また母音キーを複数プレスしなければならない場合にプレスの順序が不正ならおやつが取れない工夫を施します。その後は以下のステップでおやつの数を徐々に少なくして行きます。

(1) 母音キー以外の所要キーのおやつ無し。これらを LED ライトで光らせる。
(2) 母音キーのおやつ無し。これも LED ライトで光らせる。カナ文字を聞かせ、そのカナ文字を放音したらおやつを与える。
(3)LED ライト無し。カナ文字を聞かせ、そのカナ文字を放音したらおやつを与える。
(4) おやつを見せてその名称"おやつ"を聞かせる。"おやつ"を放音したらおやつを与える。

　教える言葉のカテゴリーは以下のような順とします。

(1) カナ文字
(2) 外界の物理的存在・状態・事象
(3) 数、順番、時間、年齢、性別、氏名
(4) 欲求、意図、主張
(5) 感想、評価、推測

例えば (2)、(3) に関しては以下のような言葉を教えます。

・おやつ、わたし、あなた、あつい、さむい、ひとつ、ふたつ、さいご、あした

　言葉を学習する動物は以下の条件を満たさなければなりません。

・知能、記憶能力が高い
・知的好奇心、知識欲がある
・自制心がある
・温和で落ち着きがある
・雌である

さらに、群れで生活する動物においては以下のシチュエーションのいずれかを選択しなければなりません。

・群れから隔離する
・群れから隔離しない

どの動物に言語教育を行うか、群れから隔離するかしないか等は、AI アニマルでシミュレーションを行ってデータを収集してから判断するのが賢明でしょう。
　もし動物が人工音声を用いて人間と日常会話を交わすまで上達したら手話を教えます。ただし、動物は両腕・十指を有し、かつそれらを器用に動かすことができなければなりません。

8. プログラム
　プログラムは linux、BSD の Xlib で動きます。

・Slackware15、jammypup64_9.8、netBSD10RC4(放音無し)

図 1 の構成の場合は市販のハードウェアがそのまま使えそうです。MIDI 鍵盤を USB 接続して使用する場合のソフトウェアはかなり流布しているようです。その主な修正点はピッチ、note on、note off の情報源と note on、note off の組み合わせです。
　`espk_af` が 0 なら espeak-ng を、`espk_af` が 1 なら音楽プレーヤーを使用します。

```
#define espk_af 0/*1*/
#define Audio "espeak-ng"
#define X_PATH "/root/gcc/v_mp3"
#define FT "mp3"
```

・ja_dict：47652 bytes

`espk_af` が 0 でも"ん"には音楽プレーヤーを使用します。`Audio`、`X_PATH`、`FT` は各々その名称、音声ファイルの所蔵先、その拡張子です。

```
#define espk_af /*0*/1
#define AUDIO "mpg321"
#define X_PATH "/root/gcc/v_mp3"
#define FT "mp3"
```

・びゅーん：[6][2][2][3][2][1][1][5].mp3

上記の音声ファイルにおいてピッチの 5 はディスプレイの p 矩形列の 4 に対応しています。
　結合かなとピッチをターミナルに出力したくない場合は以下のようにします。

```
#define kanaDATA 0
```

　実行ファイル名の先頭が@であれば upper タイプとなります。

・upper タイプ：@add10
・lower タイプ：add10

　設定ファイル blecfg は/root に置きます。lower タイプにおいて C or V を採用する場合は、設定ファイル blecfg において、以下のようにします。

```
no.17::2    /* lt1 */
```

lower タイプにおいて'.' or ',' を採用する場合は以下のようにします。

```
no.18::2    /* rt1 */
```

プロセスの kill 処理を行わない場合は以下のようにします。

```
no.26::0    /* break */
```

　日本語キーは無変換、変換、カタカナ・ひらがな・ローマ字ですが、これらの二つを同時にプレスすると disorder になる可能性があります。したがって、母音キー以外の所要キーの数が多い場合は、それらのプレスを二分割しなければならないケースがあります。このプレスの二分割は日本語 106 キーボードの配線を物理的に swap すれば回避することができます。

・swap：左 Alt $\Longleftrightarrow$ 無変換
・swap：右 Alt $\Longleftrightarrow$ 変換
・swap：右 Ctrl $\Longleftrightarrow$ カタカナ・ひらがな・ローマ字

これらの swap を行ったら

```
no.25::1    /* ideal */
```
とします。これによりキャンセルキーも swap され、その位置は不変です。
配線の物理的 swap を行い、設定ファイル blecfg を修正すると、無変換からカタカナ・ひらがな・ローマ字までのキーの割り当ては以下のように一部変化します。

・前："っ"、"ー"、"ん"、shift
・後："っ"、"ん"、"ー"、shift

****

List 1:add10.c

```
/* add10.c                                                              */
/* Kikuchi Morio 2024.4.1                                               */
/* WINDOW SYSTEM:X                                                      */
/* COMPILER:gcc 11.2.0                                                  */
/* COMMANDLINE:gcc -w dr.c -I/usr/include/X11                           */
/*                 -o dr -L/usr/lib -lX11 -lm                           */

#define WX 1
#define Kana 1
#define kanaDATA /*0*//*1*//*2*//*3*/1

#if !WX
#else
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/Xlocale.h>
#include <X11/cursorfont.h>
#include <X11/keysym.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#define espk_af /*0*//*1*/0
#if espk_af==0
#define AUDIO "espeak-ng"
#define Audio "mpg321"
/*#define Audio "ogg123"*/
#elif espk_af==1
#define AUDIO "mpg321"
/*#define AUDIO "ogg123"*/
#else
#define AUDIO "_add2_kana"
#endif

#define X_PATH "/root/gcc/v_mp3"
#define FT "mp3"
/*#define X_PATH "/root/gcc/v_ogg"
#define FT "ogg"*/

#define psSPC 4

#define TRANS (65535./255)
#define UTF8 1                      /* 0, 1 */

#if UTF8
#define DKK 3
#else
#define DKK 2
#endif
```

```
#define SZ DKK                      /* 3:UTF8 */
#endif

#define AS 5

#define XDSDY dv
#define dbl double
#define READ 0
#define WRITE 1
#define DOSFONT 1

#define VGACOLORS 16

#define UDX 9
#define UDY 18
#define UdX 8                       /* stc_dosfont */
#define UdY 16                      /* stc_dosfont */
#define ds 60
#define ftos itos

char mapping;
int refill,XRESO,YRESO,/*YRESO_,*/WB,CLMN,ROW,dv,putperiod,kp,wx,wy;
int BPv,BPs,BPa,BPm,BPi,BPd,BPlt1,BPrt1,BPha,BPdaha,BPaya,BPrtn,BPclr,BPideal,BPbreak;
int DX_FRAME,DY_CAPTION,DY_FRAME;
int L1,L2,L3,L4,L5,L6,L7,L8,L9,L10,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10;
int L11,R11;
int l1,r1;
int cnt;
dbl readcfg[30+1];
long fsize_g,xi;
dbl sec[2],ksum,pitch[10];

int argc_;
char **argv_,mf[4];

unsigned char *p1,*p2,*sjs,*utf8,*euc;
#if Kana
char kn[4][50*SZ];
FILE *fpk,*fpt;
#endif

#if !WX
#else
typedef struct {
int back,fore;} bf;
bf bfset[]={{15,0},{0,15}};

XColor irgb[VGACOLORS];

Display *d;
int screen,depth;
Colormap cmap;
Window rw,w;
XSizeHints sh;
```

```
GC gcdisplay;
Pixmap pmap1;
XFontSet font_fs;
XFontStruct **info;
Cursor cursor;
XEvent event;
KeySym keysym;
Visual *vis;
XImage *image;
Atom atm1,atm2;

char **flist,**mlist,*def;
int mcount;
#endif

void setup(void),restore_3(void),bitblt(char,int,int,int,int,int,int),BitBlt_full(void),
     setstccolor(int),cleardevice_(char,int,int,int,int),initpalette(void),kbhit_(void),
     closegraph_(void),line(char,dbl,dbl,dbl,dbl,int),lines(char,int),get_pitch(void),
     get_type(void),putstr(int,int,int,char *);
char stc_dosfont(char,int,int,unsigned char *,int);
char *itos(int);
int initgraph_(void),ff_fc(dbl),stc(char,int,int,unsigned char *,int);

#if !WX
#else
int wndproc(void);
#endif



int main(int argc,char **argv)
{
char stmp[SZ*3+1];

#if Kana
get_pitch();
#endif

argc_=argc;
argv_=(char **)argv;
/*printf("%s\n",argv_[0]);*/

#if Kana
#if !WX
#else
dv=-2;
#endif
#endif

WB=0;
refill=1;

#if DOSFONT
if(set_1B_3B()) {printf(" files not found\n");return 1;}
#endif
```

```
if(initgraph_()==1) return 1;
get_type();

#if Kana
cleardevice_(1,0,0,XRESO,YRESO);
lines(1,13);
/*WB=0*/
#endif

printf("あ行\n");

if(BPi){
if(!BPha) strcpy(stmp,"あ行");
else      strcpy(stmp,"は行");
putstr(2,1,ROW+1,stmp);
}

while(1){
kbhit_();
if(refill==0) break;
}

closegraph_();

return 0;
}/** main **/



void delay_(long millisecond)
{
long oldtime,nowtime,dtime;
dbl i=CLOCKS_PER_SEC,j;

j=millisecond;
millisecond=j*(i/1000.);
oldtime=clock();

while(1){
kbhit_();
if(refill==0) break;

nowtime=clock();dtime=nowtime-oldtime;
if(dtime>=millisecond) break;
if(dtime<0) break;
}
}/** delay_ **/



void get_pitch(void)
{
int i,j,k,l,m,n,o,p,q,z;
dbl dlt=7.981869,val[2];
dbl mpl[9]={1, 1.1224, 0.5450, 2.1811, 1.1224, 1.1224, 0.5450, 2.1811, 1.1224};

val[1]=0;                           /* kp:0 */
```

```
pitch[0]=0;                          /* Do */
for(z=1;z</*9*/10;z++){
     if(z==1) i=j=k=l=m=n=o=p=q=0;    /* kp:1 */
else if(z==2) {i=0;j=1;k=l=m=n=o=p=q=0;}
else if(z==3) {i=0;j=1;k=2;l=m=n=o=p=q=0;}
else if(z==4) {i=0;j=1;k=2;l=3;m=n=o=p=q=0;}
else if(z==5) {i=0;j=1;k=2;l=3;m=4;n=o=p=q=0;}
else if(z==6) {i=0;j=1;k=2;l=3;m=4;n=5;o=p=q=0;}
else if(z==7) {i=0;j=1;k=2;l=3;m=4;n=5;o=6;p=q=0;}
else if(z==8) {i=0;j=1;k=2;l=3;m=4;n=5;o=6;p=7;q=0;}
else if(z==9) {i=0;j=1;k=2;l=3;m=4;n=5;o=6;p=7;q=8;}  /* kp:9 */

val[0]=mpl[i]*mpl[j]*mpl[k]*mpl[l]*mpl[m]*mpl[n]*mpl[o]*mpl[p]*mpl[q]*dlt;
/*printf("dp:%f\n",16.02*val[0]/dlt);*/

val[1]+=val[0];
pitch[z]=val[1];
}

/*printf("\nsum:%f\n",val[1]);
printf("dlt_:%f\n",dlt*99/val[1]);*/
}/** get_pitch **/



void get_type(void)
{
int len,i,slash;

len=strlen(argv_[0]);

#if !WX
#else
if(/*argv_[0][len-1]=='h'*/0){
R6=XK_Hiragana_Katakana;
R11=XK_Henkan;
printf("shift key for HA line ～ WA line:XK_Henkan\n");
}
else{
R6=XK_Henkan;
R11=XK_Hiragana_Katakana;
printf("shift key for HA line ～ WA line:XK_Hiragana_Katakana\n");
}

i=len-1;
while(1){
if(argv_[0][i]=='/') {slash=i;break;}
if(i==0) {slash=-1;break;}

i--;
}

if(argv_[0][0]=='@' || (slash>-1 && argv_[0][slash+1]=='@')){  /* upper */
printf("upper\n");

/* upper */
```

```
L1='f';
l1='f';
L2='r';
L3='e';
L4='w';
L5='q';

L6=' ';
L7='5';
L8='4';
L9='3';
L10='2';

L11=XK_Muhenkan;

R1='l';
r1='l';
R2='o';
R3='p';
R4='@';
R5='[';

if(!BPdaha){
R7='9';
R8='0';
}
else{
R7='0';
R8='9';
}
if(!BPaya){
R9='-';
R10='^';
}
else{
R9='^';
R10='-';
}
}
else{                                   /* lower */
printf("lower\n");

/* lower */
L1='c';
l1='v';
L2='f';
L3='d';
L4='s';
L5='a';

L6=' ';
L7='r';
L8='e';
L9='w';
L10='q';
```

```
L11=XK_Muhenkan;

R1='.';
r1=',';
R2='l';
R3=';';
R4=':';
R5=']';

if(!BPdaha){
R7='o';
R8='p';
}
else{
R7='p';
R8='o';
}
if(!BPaya){
R9='@';
R10='[';
}
else{
R9='[';
R10='@';
}
}
#endif
}/** get_type **/


void lines(char flag,int color)
{
int i,dlt,n,dy;

if(BPi)
dlt=UDY*ROW+(UDY*2.5+4);
else
dlt=UDY*ROW+(UDY*2.5+4-(UDY+8));
dy=1;
n=4-dy;

WB=1;
i=3;
cleardevice_(1,ds*i,dlt+ds*(n-2),ds,ds*2);
i=7;
cleardevice_(1,ds*i,dlt+ds*(n-2),ds,ds*2);
WB=0;

stc(1,0+UdX*1,dlt+19,"s",1);
stc(1,0+UdX,dlt+ds*(2-dy)+19,"p",1);

/* indicator */
if(BPi){
putperiod=4;
```

```
line(flag,0,dlt-UDY-8,ds*(8+1),dlt-UDY-8,color);
putperiod=0;

i=1;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
i=2;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
line(flag,ds*i-1,dlt-UDY-8,ds*i-1,dlt-UDY-8+UDY+8,color-1);
i=3;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
i=4;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
line(flag,ds*i-1,dlt-UDY-8,ds*i-1,dlt-UDY-8+UDY+8,color-1);
i=5;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
i=6;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
line(flag,ds*i-1,dlt-UDY-8,ds*i-1,dlt-UDY-8+UDY+8,color-1);
i=7;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
i=8;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
i=9;
line(flag,ds*i-0,dlt-UDY-8,ds*i-0,dlt-UDY-8+UDY+8,color-1);
line(flag,ds*i-1,dlt-UDY-8,ds*i-1,dlt-UDY-8+UDY+8,color-1);
}

/*999*/
putperiod=4;
line(flag,ds*9,YRESO-1-ds*3-UDY-7,/*XRESO-1*/ds*10,YRESO-1-ds*3-UDY-7,color);
putperiod=0;

line(flag,0,dlt,/*XRESO-1*/ds*10,dlt,color);
line(flag,0,dlt+ds*(n-2),/*XRESO-1*/ds*10,dlt+ds*(n-2),color);
line(flag,0,dlt+ds*n-1,/*XRESO-1*/ds*10,dlt+ds*n-1,color);

line(flag,ds*10,dlt+ds*(n-/*2*/3),ds*10,dlt+ds*n-1,color);

for(i=1;i<9;i++){
line(flag,ds*i,dlt,ds*i,dlt+ds*n,color);
if(i==3+1 || i==7+1) WB=1;
else                 WB=0;
setstccolor(bfset[WB].fore);
stc(1,ds*(i-1)+UdX,dlt+ds*n-UdX,ftos(i-1),1);
}
WB=0;
i=9;
line(flag,ds*i,dlt+ds*(n-/*2*/3),ds*i,dlt+ds*n,color);
setstccolor(bfset[WB].fore);
stc(1,ds*(i-1)+UdX,dlt+ds*n-UdX,ftos(i-1),1);
i=10;
stc(1,ds*(i-1)+UdX,dlt+ds*n-UdX,ftos(i-1),1);
}/** lines **/
```

```
int set_1B_3B(void)
{
unsigned char buf[3],buf_[3];
long i,size,size_;
FILE *fp_fnt1,*fp_fnt2,*fp_sjs,*fp_euc;
FILE *fp_nkf,*fp_sjs_,*fp_utf8_,*fp_utf8;

#if /*UTF8*/0
fp_sjs=fopen("SJS.BIN","rb");
if(fp_sjs==NULL) return 1;
fseek(fp_sjs,0,2);
size=ftell(fp_sjs);
fseek(fp_sjs,0,0);

fp_sjs_=fopen("sjis_.bin","wb");
fp_utf8_=fopen("utf8.bin","wb");

for(i=0;i<size/2;i++){
fread(buf,1,2,fp_sjs);              /* 2 bytes */
fp_nkf=fopen("nkf_sjis.bin","wb");
fwrite(buf,1,2,fp_nkf);
fclose(fp_nkf);

system("nkf -S -w nkf_sjis.bin > nkf_utf8.bin");
fp_nkf=fopen("nkf_utf8.bin","rb");
fseek(fp_nkf,0,2);
size_=ftell(fp_nkf);
if(size_==3){
fseek(fp_nkf,0,0);
fread(buf_,1,/*2*/3,fp_nkf);

fwrite(buf,1,2,fp_sjs_);            /* 2 bytes */
fwrite(buf_,1,/*2*/3,fp_utf8_);     /* 3 bytes */
}/**if(size)**/

fclose(fp_nkf);
}/**for(i)**/

fclose(fp_sjs);
fclose(fp_sjs_);
fclose(fp_utf8_);

system("cp -a sjis_.bin sjis.bin");
unlink("nkf_sjis.bin");
unlink("nkf_utf8.bin");

return 1;
#endif



fp_fnt1=fopen("/root/gcc/Tmp/DOSFONT1.BIN","rb");
if(fp_fnt1==NULL) return 1;
fseek(fp_fnt1,0,2);
size=ftell(fp_fnt1);
```

```
p1=(unsigned char *)malloc(size);
fseek(fp_fnt1,0,0);
for(i=0;i<size;i++){
if(fread(buf,1,1,fp_fnt1)<1) printf(" ?\n");
p1[i]=buf[0];
}
fclose(fp_fnt1);

fp_fnt2=fopen("/root/gcc/Tmp/DOSFONT2.BIN","rb");
if(fp_fnt2==NULL) return 1;
fseek(fp_fnt2,0,2);
size=ftell(fp_fnt2);
p2=(unsigned char *)malloc(size);
fseek(fp_fnt2,0,0);
for(i=0;i<size;i++){
if(fread(buf,1,1,fp_fnt2)<1) printf(" ?\n");
p2[i]=buf[0];
}
fclose(fp_fnt2);

fp_sjs=fopen("/root/gcc/Tmp/sjis.bin","rb");
if(fp_sjs==NULL) return 1;
fseek(fp_sjs,0,2);
size=ftell(fp_sjs);
sjs=(unsigned char *)malloc(size);
fseek(fp_sjs,0,0);
for(i=0;i<size;i++){
if(fread(buf,1,1,fp_sjs)<1) printf(" ?\n");
sjs[i]=buf[0];
}
fclose(fp_sjs);

#if UTF8
fp_utf8=fopen("/root/gcc/Tmp/utf8.bin","rb");
if(fp_utf8==NULL) return 1;
fseek(fp_utf8,0,2);
size=ftell(fp_utf8);
fsize_g=size;
utf8=(unsigned char *)malloc(size);
fseek(fp_utf8,0,0);
for(i=0;i<size;i++){
if(fread(buf,1,1,fp_utf8)<1) printf(" ?\n");
utf8[i]=buf[0];
}
fclose(fp_utf8);
#else
fp_euc=fopen("/root/gcc/Tmp/euc.bin","rb");
if(fp_euc==NULL) return 1;
fseek(fp_euc,0,2);
size=ftell(fp_euc);
fsize_g=size;
euc=(unsigned char *)malloc(size);
fseek(fp_euc,0,0);
for(i=0;i<size;i++){
if(fread(buf,1,1,fp_euc)<1) printf(" ?\n");
```

```
euc[i]=buf[0];
}
fclose(fp_euc);
#endif

return 0;
}/** set_1B_3B **/


void putpixel(char flag,int x,int y,int color)
{
#if !WX
#else
Drawable dable;

XSetForeground(d,gcdisplay,irgb[color].pixel);
if(flag==1) dable=pmap1;
else        dable=w;

XDrawPoint(d,dable,gcdisplay,x,y);
#endif
}/** putpixel **/


void line(char flag,dbl x1_,dbl y1_,dbl x2_,dbl y2_,int color)
{
int x1,y1,x2,y2,dx,dy,x,y;
int c,d,e,sx,sy;
int putflag=1,putcount=0;

x1=ff_fc(x1_);y1=ff_fc(y1_);
x2=ff_fc(x2_);y2=ff_fc(y2_);

dx=abs(x2-x1);
dy=abs(y2-y1);

if(x1<=x2) sx=1;
else       sx=-1;
if(y1<=y2) sy=1;
else       sy=-1;

if(dx>=dy){
c=2*dy;d=2*(dy-dx);e=c-dx;

x=x1;
y=y1;

while(1){
if(putperiod==0) ;
else{
if(putflag==1) ;
putcount++;
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
```

```
}
}

if(putperiod==0 || putflag==1) putpixel(flag,x,y,color);

if(e<0) e+=c;
else    {e+=d;y+=sy;}

x+=sx;
if(sx>=0) {if(x>x2) break;}
else      {if(x<x2) break;}
}
}/**if(dx,dy)**/
else{
c=2*dx;d=2*(dx-dy);e=c-dy;

x=x1;
y=y1;

while(1){
if(putperiod==0) ;
else{
if(putflag==1) ;
putcount++;
if(putcount==putperiod){
putcount=0;
if(putflag==1) putflag=0;else putflag=1;
}
}

if(putperiod==0 || putflag==1) putpixel(flag,x,y,color);

if(e<0) e+=c;
else    {e+=d;x+=sx;}

y+=sy;
if(sy>=0) {if(y>y2) break;}
else      {if(y<y2) break;}
}
}/**else(dx,dy)**/
}/** line **/


void ppixel(char flag,int nx,int ny,int pcolor)
{
#if !WX
#else
Drawable dable;

if(flag==1) dable=pmap1;
else        dable=w;

XDrawPoint(d,dable,gcdisplay,nx-1,ny-(UdY-2));  /* W:ny, L:ny-(UdY-2) */
#endif
}/** ppixel **/
```

```
unsigned char array_(char flag,int nx,int ny,unsigned char val,char bytes,char dlt)
{
unsigned long Offset;

if(nx<0 || nx>0xff || ny<0 || ny>0xff) return 0;

if(flag<2)
Offset=ny*bytes+dlt;
else
Offset=(unsigned long)nx*256*bytes+ny*bytes+dlt;

if(flag%2==0){                      /* read */
if(flag==0) val=p1[Offset];
else        val=p2[Offset];
}
else{                               /* write */
if(flag==1) p1[Offset]=val;
else        p2[Offset]=val;
}

return val;
}/** array_ **/


unsigned char work_Rect(char RW,int x,int y,
               unsigned char pcolor,char dlt)
{
unsigned char val;

if(!RW){                            /* read */
val=array_(READ+2,x,y,-1,32,dlt);
}
else{                               /* write */
array_(WRITE+2,x,y,pcolor,32,dlt);
}

return val;
}/** work_Rect **/


unsigned char work_p(char RW,int x,int y,
            unsigned char pcolor,char dlt)
{
unsigned char val;

if(!RW){                            /* read */
val=array_(READ,x,y,-1,16,dlt);
}
else{                               /* write */
array_(WRITE,x,y,pcolor,16,dlt);
}

return val;
```

```
}/** work_p **/


char bit_read(unsigned char the_byte,unsigned char dlt)
{
static unsigned char val[1];

val[0]=the_byte;

if(/*!RW*/1){                           /* read */
if(dlt) {val[0]=val[0] >> dlt;}
val[0]=val[0] & 1;
return val[0];
}
else{                                /* write */
return 0;
}
}/** bit_read **/


char stc_dosfont(char flag,int x,int y,unsigned char *str,int size)
{
char pxl;
unsigned char str_[72];
int i,j,end;

if(size==1){
for(j=0;j/*<=15*/<UdY;j++){
str_[j]=work_p(READ,0x00,str[0]-0x20,-1,j);
/*printf(" %d",str_[j]);*/
}

for(j=0;j/*<=15*/<UdY;j++)
for(i=0;i</*8*/UdX;i++){
pxl=bit_read(str_[j],i);
if(pxl) ppixel(flag,x+/*8*/UdX-i,y+j,-1);
}
}/**if(size)**/
else{
#if UTF8
end=fsize_g-1-2;
for(i=0;i<=end;i+=/*2*/3){
if(utf8[i]==str[0] && utf8[i+1]==str[1] && utf8[i+2]==str[2]){
str[0]=sjs[i*2/3];
str[1]=sjs[i*2/3+1];
break;
}
}

if(i>end) return 1;
#else
end=fsize_g-1-1;
for(i=0;i<=end;i+=2){
if(euc[i]==str[0] && euc[i+1]==str[1]){
str[0]=sjs[i];
```

```
str[1]=sjs[i+1];
break;
}
}

if(i>end) return 1;
#endif

for(j=0;j/*<=31*/<UdY*2;j++){
str_[j]=work_Rect(READ,str[0],str[1],-1,j);
/*printf(" %d",str_[j]);*/
}

for(j=0;j/*<=30*/<UdY*2-1;j+=2){
for(i=0;i</*8*/UdX;i++){
pxl=bit_read(str_[j],i);
if(pxl) ppixel(flag,x+/*8*/UdX-i,y+j/2,-1);
}

for(i=0;i</*8*/UdX;i++){
pxl=bit_read(str_[j+1],i);
if(pxl) ppixel(flag,x+/*8+8*/UdX*2-i,y+j/2,-1);
}
}/**for(j)**/
}/**else(size)**/

return 0;
}/** stc_dosfont **/


int stc(char flag,int x,int y,unsigned char *str,int size)
{
int over_end=0;

#if UTF8

#if DOSFONT
over_end=stc_dosfont(flag,x,y+XDSDY,str,size);
#else
/*Xutf8DrawString(d,dable,font_fs,gcdisplay,x,y+XDSDY,str,size)*/;
#endif

#else

#if DOSFONT
over_end=stc_dosfont(flag,x,y+XDSDY,str,size);
#else
/*XmbDrawString(d,dable,font_fs,gcdisplay,x,y+XDSDY,str,size)*/;
#endif

#endif

return over_end;
}/** stc **/
```

```
int read_cfg(char *home)
{
char onceflag;
int i,j,k,len;
char *p,str[10];
FILE *fp;

if((fp=fopen(home,"rb"))==NULL) return 1;
fseek(fp,0,2);len=ftell(fp);
fseek(fp,0,0);

p=(char *)malloc(len);
fread(p,1,len,fp);
/*printf("%c\n",p[0]);*/

i=0;
while(1){
     if(i==len-1-6) {printf("no\n");break;}  /* no left */
else if(i==len-1-7) {/*printf("one 0x%02x\n",p[i+7]);*/break;} /* one left */
else if(i==len-1-8){                /* two left */
if(p[i]=='n' && p[i+1]=='o' && p[i+2]=='.'){
if((p[i+3]==' ' || (p[i+3]>=0x30 && p[i+3]<=0x39)) && (p[i+4]>=0x30 && p[i+4]<=0x39)){
str[0]=p[i+3];str[1]=p[i+4];str[2]='\0';
j=atoi(str);

if(p[i+5]==':' && p[i+6]==':' && (p[i+7]>=0x30 && p[i+7]<=0x39)){
str[0]=p[i+7];str[1]='\0';
readcfg[j]=atoi(str);
break;
}
}/**if(p[i+3],p[i+4])**/
}/**if('n','o','.')**/
}/**else if(i)**/
else{
if(p[i]=='n' && p[i+1]=='o' && p[i+2]=='.'){
if((p[i+3]==' ' || (p[i+3]>=0x30 && p[i+3]<=0x39)) && (p[i+4]>=0x30 && p[i+4]<=0x39)){
str[0]=p[i+3];str[1]=p[i+4];str[2]='\0';
j=atoi(str);

if(p[i+5]==':' && p[i+6]==':'){

if((p[i+7]>=0x30 && p[i+7]<=0x39) || p[i+7]=='.'){  /* no '-' */
onceflag=0;
k=0;
while(1){
if(!onceflag) {if((p[i+7+k]<0x30 && p[i+7+k]!='.') || p[i+7+k]>0x39) break;}
else          {if(p[i+7+k]<0x30 || p[i+7+k]>0x39) break;}
if(k==1 && p[i+7+0]=='0' && p[i+7+1]=='.') onceflag=1;
else if(k==0 && p[i+7+0]=='.') onceflag=1;
k++;
if(i+7+k==len-1) break;             /* when LF */
}

strncpy(str,&p[i+7],k);
```

```
str[k]='\0';
readcfg[j]=atof(str);
/*printf("str:%s\n",str);*/
}
else if(p[i+7]=='-' && ((p[i+8]>=0x30 && p[i+8]<=0x39) || p[i+7]=='.')){
k=0;
onceflag=0;
while(1){
if(!onceflag) {if((p[i+7+1+k]<0x30 && p[i+7+1+k]!='.') || p[i+7+1+k]>0x39) break;}
else          {if(p[i+7+1+k]<0x30 || p[i+7+1+k]>0x39) break;}
if(k==2 && p[i+7+1]=='0' && p[i+7+2]=='.') onceflag=1;
else if(k==1 && p[i+7+1]=='.') onceflag=1;
k++;
if(i+7+1+k==len-1) break;           /* when LF */
}

strncpy(str,&p[i+7],k+1);
str[k+1]='\0';
readcfg[j]=atof(str);
/*printf("str:%s\n",str);*/
}

}/**if(p[i+5],p[i+6])**/
}/**if(p[i+3],p[i+4])**/
}/**if('n','o','.')**/
}/**else(i)**/

i++;if(i>len-1) break;
}/**while(1)**/

free(p);
fclose(fp);

return 0;
}/** read_cfg **/



void setup(void)
{
char flag;
int n,dy,xreso[2];

dy=1;
n=4-dy;

CLMN=56;
ROW=12;
BPv=13;
BPs=1+25*7;
BPa=200;
BPm=2;                              /* Alt or Ctrl */
BPi=1;
BPd=25;
BPlt1=0;
BPrt1=0;
```

```
BPha=0;
BPdaha=0;
BPaya=0;
BPrtn=1;                            /* 1:a, 2:ya */
BPclr=0;
BPideal=0;
BPbreak=0;

readcfg[1]=CLMN;
readcfg[2]=ROW;
readcfg[3]=BPv;
readcfg[4]=BPs;
readcfg[7]=BPa;
readcfg[10]=BPm;                    /* mask */
readcfg[12]=BPi;                    /* indicator */
readcfg[16]=BPd;
readcfg[17]=BPlt1;
readcfg[18]=BPrt1;
readcfg[19]=BPha;
readcfg[20]=BPdaha;
readcfg[21]=BPaya;
readcfg[22]=BPrtn;
readcfg[24]=BPclr;
readcfg[25]=BPideal;
readcfg[26]=BPbreak;

if(!read_cfg("/root/blecfg")){
CLMN=readcfg[1];
ROW=readcfg[2];
BPv=readcfg[3];
BPs=readcfg[4];
BPa=readcfg[7];
BPm=readcfg[10];
BPi=readcfg[12];
BPd=readcfg[16];
BPlt1=readcfg[17];
BPrt1=readcfg[18];
BPha=readcfg[19];
BPdaha=readcfg[20];
BPaya=readcfg[21];
BPrtn=readcfg[22];
BPclr=readcfg[24];
BPideal=readcfg[25];
BPbreak=readcfg[26];

flag=1;
}
else flag=0;

if(BPv<0) BPv*=-1;

if(BPv>9){
strcpy(mf,"f");
if(BPv==10) BPv=11;
strcat(mf,ftos(BPv-10));
```

```
}
else{
strcpy(mf,"m");
if(BPv==0) BPv=1;
strcat(mf,ftos(BPv));
}

if(flag)
printf("readcfg  v:%d(%s) s:%d m:%d d:%d\n",BPv,mf,BPs,BPm,BPd);
else
printf("no readcfg  v:%d(%s) s:%d m:%d d:%d\n",BPv,mf,BPs,BPm,BPd);

if(CLMN%2) CLMN--;
if(CLMN<8) CLMN=8;
if(ROW<2) ROW=2;
printf("CLMN:%d ROW:%d\n",CLMN,ROW);
kp=4;
printf("s:%d kp:%d\n",BPs,kp);

if(BPv>9){
strcpy(mf,"f");
strcat(mf,itos(BPv-10));
}
else{
strcpy(mf,"m");
strcat(mf,itos(BPv));
}

xreso[0]=ds*(8+2)+DX_FRAME*2+1;
#if !WX
#else
xreso[1]=8+CLMN*UdX+UdX*6+DX_FRAME*2-3;
#endif

if(xreso[1]<xreso[0]) XRESO=xreso[0];
else                  XRESO=xreso[1];

if(BPi)
YRESO=UDY*ROW+(UDY*2.5+4)+ds*n/*+DY_CAPTION+DY_FRAME*2*/;
else
YRESO=UDY*ROW+(UDY*2.5+4-(UDY+8))+ds*n/*+DY_CAPTION+DY_FRAME*2*/;
/*YRESO_=YRESO-ds*n;*/
}/** setup **/


int ff_fc(double val_d)
{
int val_i,val;

val_i=floor(val_d);
val=(val_d-val_i<0.5)?val_i:(val_i+1);

return val;
}/** ff_fc **/
```

```
void restore_3(void)
{
if(/*ftp==0 && */refill==0) goto skip;

bitblt(1,0,0,XRESO,YRESO,0,0);

skip:{}
}/** restore_3 **/


void BitBlt_full(void)
{
bitblt(1,0,0,XRESO,YRESO,0,0);
}/** BitBlt_full **/


void initpalette(void)
{
int i;

irgb[0].red=0;irgb[0].green=0;irgb[0].blue=0;

irgb[9].red=0;irgb[9].green=0;irgb[9].blue=255;  /* blue */
irgb[10].red=0;irgb[10].green=255;irgb[10].blue=0;  /* green */
irgb[11].red=0;irgb[11].green=255;irgb[11].blue=255;  /* cyan */
irgb[12].red=255;irgb[12].green=0;irgb[12].blue=0;  /* red */
irgb[13].red=255;irgb[13].green=0;irgb[13].blue=255;  /* magenta */
irgb[14].red=255;irgb[14].green=255;irgb[14].blue=0;  /* yellow */

irgb[15].red=255;irgb[15].green=255;irgb[15].blue=255;

for(i=1;i<7;i++){                    /* 1 -> 6 */
if(irgb[9+(i-1)].red==255)
irgb[i].red=127+64;
if(irgb[9+(i-1)].green==255)
irgb[i].green=127+64;
if(irgb[9+(i-1)].blue==255)
irgb[i].blue=127+64;
}

for(i=7;i<9;i++){                    /* 7, 8 */
irgb[i].red=127+32*(8-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}

#if WX
for(i=0;i<VGACOLORS;i++){
irgb[i].red=ff_fc(irgb[i].red*TRANS);
irgb[i].green=ff_fc(irgb[i].green*TRANS);
irgb[i].blue=ff_fc(irgb[i].blue*TRANS);
}

for(i=0;i<VGACOLORS;i++)
```

```
XAllocColor(d,cmap,&irgb[i]);
#endif
}/** initpalette **/


#if !WX
#else
int initgraph_(void)
{
if((d=XOpenDisplay(""))==NULL) return 1;
screen=DefaultScreen(d);

cmap=DefaultColormap(d,screen);
initpalette();

setup();

rw=DefaultRootWindow(d);
w=XCreateSimpleWindow(d,rw,80*7,0,XRESO,YRESO,0,
                      irgb[bfset[WB].back].pixel,
                      irgb[bfset[WB].back].pixel);

sh.flags=PPosition | PSize;
sh.x=0;sh.y=0;
sh.width=XRESO;sh.height=YRESO;
XSetStandardProperties(d,w,"SS","SS",None,argv_,argc_,&sh);

atm1=XInternAtom(d,"WM_PROTOCOLS",False);
atm2=XInternAtom(d,"WM_DELETE_WINDOW",False);
XSetWMProtocols(d,w,&atm2,1);

XSelectInput(d,w,KeyPressMask | PointerMotionMask | ExposureMask |
                 EnterWindowMask | LeaveWindowMask | ButtonPress |
                 StructureNotifyMask | SubstructureNotifyMask);
XMapWindow(d,w);

gcdisplay=XCreateGC(d,w,0,NULL);

depth=DefaultDepth(d,screen);
pmap1=XCreatePixmap(d,w,XRESO,YRESO,depth);  /* visual page */

cursor=XCreateFontCursor(d,XC_arrow);
XDefineCursor(d,w,cursor);

XSetLineAttributes(d,gcdisplay,/*2*/1,LineSolid,CapButt,JoinMiter);

/*initIME();*/
     if(0) font_fs=XCreateFontSet(d,"-*-*-medium-r-normal--14-*-*-*-*-*-*-*",&mlist,&mcount,&def);
else if(0) font_fs=XCreateFontSet(d,"-*-*-medium-r-semicondensed--13-*-*-*-*-*-*-*",&mlist,&mcount,&def);
else if(0) font_fs=XCreateFontSet(d,"6x13",&mlist,&mcount,&def);
if(0) XFontsOfFontSet(font_fs,&info,&flist);

return 0;
}/** initgraph_ **/
```

```
void closegraph_(void)
{
#if DOSFONT
free(p1);
free(p2);
free(sjs);
#if UTF8
free(utf8);
#else
free(euc);
#endif
#endif

if(0) XFreeFontSet(d,font_fs);
XFreeColormap(d,cmap);
XFreeCursor(d,cursor);
XFreePixmap(d,pmap1);
XFreeGC(d,gcdisplay);

XDestroyWindow(d,w);/*XFlush(d);*/
XCloseDisplay(d);
}/** closegraph_ **/


void cleardevice_(char flag,int x,int y,int xsize,int ysize)
{
XSetForeground(d,gcdisplay,irgb[bfset[WB].back].pixel);

if(flag==0)
XFillRectangle(d,w,gcdisplay,x,y,xsize,ysize);
else if(flag==1)
XFillRectangle(d,pmap1,gcdisplay,x,y,xsize,ysize);
}/** cleardevice_ **/


void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
if(flag==0) {}
else if(flag==1)
XCopyArea(d,pmap1,w,gcdisplay,x,y,xsize,ysize,
                              x_,y_);
else if(flag==2)
XCopyArea(d,pmap1,pmap1,gcdisplay,x,y,xsize,ysize,
                                   x_,y_);

XFlush(d);
}/** bitblt **/


void setstccolor(int color)
{
XSetForeground(d,gcdisplay,irgb[color].pixel);
}/** setstccolor **/
```

```
int GKS(KeySym XK)
{
if(keysym==XK) return -1;
else return 0;
}/** GKS **/


int GKS_(long ModkeyMask)
{
if((event.xkey.state & ModkeyMask)>0) return -1;
else return 0;
}/** GKS_ **/


int GKS_m(void)
{
if(!BPm){
if(GKS_(Mod1Mask)<0) return -1;else return 0;
}
else if(BPm==1){
if(GKS_(ControlMask)<0) return -1;else return 0;
}
else{
if(GKS_(Mod1Mask)<0 || GKS_(ControlMask)<0) return -1;else return 0;
}
}/** GKS_m **/


int GKS_Lt1(void)
{
if(!BPlt1){
if(GKS(L1)<0) return -1;else return 0;
}
else if(BPlt1==1){
if(GKS(l1)<0) return -1;else return 0;
}
else{
if(GKS(L1)<0 || GKS(l1)<0) return -1;else return 0;
}
}/** GKS_Lt1 **/


int GKS_Rt1(void)
{
if(!BPrt1){
if(GKS(R1)<0) return -1;else return 0;
}
else if(BPrt1==1){
if(GKS(r1)<0) return -1;else return 0;
}
else{
if(GKS(R1)<0 || GKS(r1)<0) return -1;else return 0;
}
}/** GKS_Rt1 **/
```

```
int GKS_aya(char flag)
{
if(BPrtn==flag){
if(GKS(XK_Return)<0) return -1;else return 0;
}
else return 0;
}/** GKS_aya **/


long min(long a,long b)
{
return (a<b)?a:b;
}/** min **/


void kbhit_(void)
{
if(XPending(d)){
XNextEvent(d,&event);

if(event.type==ClientMessage &&
   event.xclient.message_type==atm1 && event.xclient.data.l[0]==atm2){
printf("Close\n");

/*if(cnt) close_espk();*/
closegraph_();
exit(1);
}
else{
wndproc();
}
}/**if(XPending(d))**/
}/** kbhit_ **/
#endif


void puts_3B(char *jis)
{
char str[5];
static int is=0,js=0;
int len,i,imax,dx,dy,DJ=1;

if(js>ROW-1){
js=ROW-1;
#if 0
bitblt(2,0,UDY,XRESO,UDY*ROW-UDY,0,0);  /* old bitblt */
cleardevice_(1,0,UDY*js,XRESO,UDY);
#else
bitblt(2,0,UDY,XRESO,UDY*ROW-UDY*0,0,0);  /* new bitblt */
cleardevice_(1,0,UDY*(js+1),XRESO,UDY);
#endif
}
```

```
len=strlen(jis);
imax=len/SZ;
/*printf("imax:%d\n",imax);*/

setstccolor(bfset[WB].fore);

for(i=0;i<imax;i++){
strncpy(str,&jis[SZ*i],SZ);str[SZ]='\0';
dx=8+UdX*is;dy=UDY*(js+DJ);
stc(1,dx,dy,str,SZ);

is+=2;
if(is>CLMN-1) {is=0;js++;}
}

/*js++;*/

/*return js;*/
}/** puts_3B **/


#if 0
void puts_1B(int js,char *jis)
{
char str[2];
int i,size,DJ=1;

size=strlen(jis);

for(i=0;i<size;i++){
strncpy(str,&jis[i],1);
str[1]='\0';
stc(1,8+UdX*(8+i),UDY*(js+DJ),str,1);
}
}/** puts_1B **/
#endif


char *itos(int i)
{
static char buf[10];

#if !WX
#else
/*gcvt(i,10,buff);*/
sprintf(buf,"%d",i);
#endif

return buf/*f*/;
}/** itos **/


int getkp(void)
{
#if !WX
```

```
#else
int rx,ry;
unsigned int mask;
Window root,child;

XQueryPointer(d,w,&root,&child,&rx,&ry,&wx,&wy,&mask);
if(wy>=YRESO-1-ds*2 && wy<=YRESO-1 && wx>-1 && wx<XRESO){
kp=wx/ds;
}
else if(wy<YRESO-1-ds*3-UDY-7 && wx>-1 && wx<XRESO){
kp=-1;
}
#endif
}/** getkp **/


void putstr(int flag,int i,int j,char *str)
{
int DJ=1,size,k,dx=1;
dbl dlt=0.5;
char stmp[SZ*3+1];

if(!flag){
cleardevice_(0,i*ds+dx,(0+j+dlt)*UDY+0,ds-(dx+1),UDY);
cleardevice_(1,i*ds+dx,(0+j+dlt)*UDY+0,ds-(dx+1),UDY);
}
else if(flag>0){
if(1) cleardevice_(0,i*ds+dx,(0+j+dlt)*UDY+0,ds-(dx+1),UDY);
cleardevice_(1,i*ds+dx,(0+j+dlt)*UDY+0,ds-(dx+1),UDY);

setstccolor(bfset[WB].fore);

size=SZ;
strcpy(stmp,str);
for(k=0;k<flag;k++){
stc_dosfont(0,i*ds+k*UdX*2+dx,(DJ+j+dlt)*UDY+XDSDY,&str[k*SZ],size);
stc_dosfont(1,i*ds+k*UdX*2+dx,(DJ+j+dlt)*UDY+XDSDY,&stmp[k*SZ],size);
}
}
else{
setstccolor(bfset[WB].fore);

size=1;flag*=-1;
for(k=0;k<flag;k++){
stc_dosfont(1,i*ds+k*UdX+dx,(DJ+j+dlt)*UDY+XDSDY,&str[k],size);
stc_dosfont(0,i*ds+k*UdX+dx,(DJ+j+dlt)*UDY+XDSDY,&str[k],size);
}
}
}/** putstr **/


void indicator(char shift,int _1ST0,char chouon,char sokuon,char nn,char knflag,char A_YA_flag)
{
int i,len;
char stmp[SZ*3+1];                  /* up to 3 kanas */
```

```
/*if(!cnt) return;
if(!BPi) return;*/

i=0;
putstr(0,i,ROW+1,stmp);

if(shift){
strcpy(stmp,"shift");               /* shift:5 characters -> -5 */
len=strlen(stmp);
putstr(-len,i,ROW+1,stmp);
}
/*else putstr(0,i,ROW+1,stmp);*/

if(/*!prtflag*/_1ST0>-1){
i=1;
putstr(0,i,ROW+1,stmp);

/*if(knflag<3){*/
if(_1ST0>-1 && _1ST0<5)
strncpy(stmp,&kn[knflag][0+SZ*AS*(_1ST0+shift*5)],SZ);
else
strncpy(stmp,&kn[knflag][0+SZ*AS*(_1ST0+0)],SZ);
stmp[SZ*1]='\0';

strcat(stmp,"行");
putstr(2,i,ROW+1,stmp);
/*}*/
}/**if(_1ST0)**/

if(knflag/*/2*/==1){
i=/*3*/2;
putstr(0,i,ROW+1,stmp);
i=/*2*/3;
strcpy(stmp,"濁");
putstr(1,i,ROW+1,stmp);
}
else if(knflag/*/2*/==2){
i=/*2*/3;
putstr(0,i,ROW+1,stmp);
i=/*3*/2;
strcpy(stmp,"半濁");
putstr(2,i,ROW+1,stmp);
}
else{
i=2;
putstr(0,i,ROW+1,stmp);
i=3;
putstr(0,i,ROW+1,stmp);
}

if(A_YA_flag==1){
i=/*5*/4;
putstr(0,i,ROW+1,stmp);
i=/*4*/5;
```

```
strcpy(stmp,"ぁ行");
putstr(2,i,ROW+1,stmp);
}
else if(A_YA_flag==2){
i=/*4*/5;
putstr(0,i,ROW+1,stmp);
i=/*5*/4;
strcpy(stmp,"や行");
putstr(2,i,ROW+1,stmp);
}
else{
i=4;
putstr(0,i,ROW+1,stmp);
i=5;
putstr(0,i,ROW+1,stmp);
}

if(chouon<0){
i=/*6*/8;
strcpy(stmp,"ー");
putstr(1,i,ROW+1,stmp);
}
else{
i=/*6*/8;
putstr(0,i,ROW+1,stmp);
}

if(sokuon){
i=/*7*/7;
putstr(0,i,ROW+1,stmp);
i=/*8*/6;
strcpy(stmp,"っ");
putstr(1,i,ROW+1,stmp);
}
else if(nn){
i=/*8*/6;
putstr(0,i,ROW+1,stmp);
i=/*7*/7;
strcpy(stmp,"ん");
putstr(1,i,ROW+1,stmp);
}
else{
i=/*7*/6;
putstr(0,i,ROW+1,stmp);
i=/*8*/7;
putstr(0,i,ROW+1,stmp);
}
}/** indicator **/


int get_boin(int _1ST0,int _1ST1)
{
int boin;
```

```
if(_1ST0==9){
     if(_1ST1==0) boin=_1ST1+1;  /* わ */
else if(_1ST1==1) boin=4+1;        /* を */
else if(_1ST1==2) boin=2+1;        /* ん */
else              boin=_1ST1+1;
}
else              boin=_1ST1+1;

return boin;
}/** get_boin **/


#if !WX
#else
void kill_PID(const char *psAP)
{
char cntflag;
int sz,spccount;
long topfpos,fpos[2];
char buf1[10],buf[10],str[32];
static char PID[10];
FILE *fp;

sz=strlen(psAP);

system("ps t > ttt");

fp=fopen("ttt","rb");
fseek(fp,0,2);
topfpos=ftell(fp);
fseek(fp,0,0);
buf1[1]='\0';buf[sz]='\0';



begin:
fread(buf1,1,1,fp);

if(!strcmp(buf1," ")){              /* ' ' */
while(1){
fread(buf1,1,1,fp);
if(strcmp(buf1," ")){               /* PID */
fpos[0]=ftell(fp)-1;
break;
}
}/**while(1)**/

while(1){
fread(buf1,1,1,fp);
if(!strcmp(buf1," ")){               /* ' ' */
fpos[1]=ftell(fp)-1;
break;
}
}/**while(1)**/
```

```
fseek(fp,fpos[0],0);
fread(PID,1,fpos[1]-fpos[0],fp);
PID[fpos[1]-fpos[0]]='\0';
fseek(fp,fpos[1]+1,0);
}/**if(!strcmp())**/
else{
fpos[0]=ftell(fp)-1;

while(1){
fread(buf1,1,1,fp);
if(!strcmp(buf1," ")){              /* ' ' */
fpos[1]=ftell(fp)-1;
break;
}
}/**while(1)**/

fseek(fp,fpos[0],0);
fread(PID,1,fpos[1]-fpos[0],fp);
PID[fpos[1]-fpos[0]]='\0';
fseek(fp,fpos[1]+1,0);
}/**else(!strcmp())**/



cntflag=/*1*/0;
spccount=1;

while(1){
if(fread(buf1,1,1,fp)<1) break;
if(ftell(fp)==topfpos) break;
if(buf1[0]==0x0a) {/*spccount=0;*/goto begin;}

     if(cntflag && !strcmp(buf1," ")){  /* ' ' */
spccount++;cntflag=0;
if(spccount==2){
/*fpos[1]=ftell(fp)-1;
fseek(fp,fpos[0],0);
fread(PID,1,fpos[1]-fpos[0],fp);
PID[fpos[1]-fpos[0]]='\0';
fseek(fp,fpos[1]+1,0);*/
}
}
else if(!cntflag && strcmp(buf1," ")){  /* !' ' */
cntflag=1;
if(spccount==1){
/*fpos[0]=ftell(fp)-1;*/
}
else if(spccount==/*4*/psSPC){
fseek(fp,-1,1);
if(fread(buf,1,sz,fp)<sz) break;
if(!strcmp(psAP,buf)) {cntflag=-1;/*break*/goto kill;}
redo:
/*else */fseek(fp,-sz+1,1);
}
}
```

```
}/**while(1)**/

kill:
if(cntflag<0){
strcpy(str,"kill -9 ");
strcat(str,PID);
system(str);
printf("%s:kill %s\n",psAP,PID);

cntflag=1;goto redo;
}

fclose(fp);
}/** kill_PID **/



int wndproc(void)
{
static char Escflag=0;
/*int length;
char buf[10];*/

#if Kana
char /*sysflag,*/addflag;
char kana[SZ*AS*10+1],buff[20],ch[]="ー",zu[]="っ",wa[]="わ";
char *pt,str[128],str2[128],stmp[SZ*3+1];
static char printflag=1,shift=0,knflag=0,A_YA_flag=0,bbflag=0,killflag=0;
static char chouon=0,sokuon=0,chousoku=0,active=1,hana=0,nn=0;
static char _4flag=0,_3flag=0,qflag=0,LCtrlflag=0;
static int _1ST[2]={0,-1},_2ND[2]={0,-1},boin=0+1,boin_b=2+1,_1ST0old=0,oldkp;
int len,i,j,i_,j_,k,l,m,n;
dbl dlt,val;
int rx,ry;
unsigned int mask;
Window root,child;
#endif

/*length=XLookupString((XKeyEvent *)&event,buf,10,&keysym,NULL);
buf[length]='\0';*/

if(event.type==KeyPress){
/*if(!mapping) goto end_Escape;*/
    if(1) XLookupString(&event,/*buf*/NULL,10,&keysym,NULL);

if(Escflag){
     if(GKS(XK_F1)<0 || GKS(XK_F2)<0) refill=0;
else Escflag=0;
goto end_Escape;
}
else{
if(GKS(XK_Escape)<0){
Escflag++;
}
}
```

```
#if Kana
if(cnt==0){
strcpy(&kn[0][0],"あいうえおかきくけこさしすせそたちつてとなにぬねのはひふへほまみむめもややゆよらりるれろわをんんん");
strcpy(&kn[1][0],"あいヴえおがぎぐげござじずぜぞだぢづでどなにぬねのばびぶべぼまみむめもややゆよらりるれろわをんんん");
strcpy(&kn[2][0],"あいうえおかきくけこさしすせそたちつてとなにぬねのぱぴぷぺぽまみむめもややゆよらりるれろわをんんん");
strcpy(&kn[3][0],"ぁぃぅぇぉかきくけこさしすせそたちってとなにぬねのはひふへほまみむめもゃゃゅゅょらりるれろわをんんん");

if(BPha/*HA*/) _1ST[0]=-1;
BitBlt_full();
/*getkp();*/
oldkp=kp;

cnt++;
if(Escflag) return 1;
}

/*999*/
     if(GKS(R8)<0) {knflag=1;/*_4flag=1;*/}  /* 濁音 */
else if(GKS(R7)<0) {knflag=2;_1ST[0]=0+5;}  /* 半濁音 */

else if(GKS_m()>=0 && GKS(L8)<0) chouon=1;  /* ー */
else if(GKS_m()<0 && GKS(L8)<0) {chousoku=1;chouon=1;}

else if(GKS(R10)<0 || GKS_aya(1)<0) A_YA_flag=1;  /* +ぁ */
else if(GKS(R9)<0 || GKS_aya(2)<0) A_YA_flag=2;   /* +や */

else if(!BPideal && GKS(L11)<0) {sokuon=1;nn=0;}  /* +っ */
else if(!BPideal && GKS(L6)<0) chouon=-1;        /* +ー */
else if(!BPideal && GKS(R6)<0)  {nn=1;sokuon=0;}  /* +ん */
else if(!BPideal && GKS(R11)<0)  shift=1;          /* kana_shift */

else if(BPideal && GKS(XK_Alt_L)<0) {sokuon=1;nn=0;}  /* +っ */
else if(BPideal && GKS(L6)<0)  {nn=1;sokuon=0;}  /* +ん */
else if(BPideal && GKS(XK_Alt_R)<0) chouon=-1;        /* +ー */
else if(BPideal && GKS(XK_Control_R)<0)  shift=1;          /* kana_shift */
#if 0
else if(GKS(L11)<0) {sokuon=1;nn=0;}  /* +っ */
else if(GKS(L6)<0) {nn=1;sokuon=0;}  /* +ん */
else if(GKS(R6)<0) chouon=-1;        /* +ー */
else if(GKS(R11)<0) shift=1;          /* kana_shift */
#endif

else if(!BPideal && !BPclr && (GKS(XK_Control_L)<0 || GKS(XK_Alt_L)<0)) goto end;
else if(!BPideal && !BPclr && (GKS(XK_Control_R)<0 || GKS(XK_Alt_R)<0)) goto end;
else if(!BPideal && BPclr && (GKS(XK_Control_L)<0 || GKS(XK_Alt_L)<0)) goto end_;
else if(!BPideal && BPclr && (GKS(XK_Control_R)<0 || GKS(XK_Alt_R)<0)){
knflag=0;
A_YA_flag=0;
}
```

```
else if(BPideal && !BPclr && (GKS(XK_Control_L)<0 || GKS(XK_Muhenkan)<0)) goto end;
else if(BPideal && !BPclr && (GKS(XK_Hiragana_Katakana)<0 || GKS(XK_Henkan)<0)) goto end;
else if(BPideal && BPclr && (GKS(XK_Control_L)<0 || GKS(XK_Muhenkan)<0)) goto end_;
else if(BPideal && BPclr && (GKS(XK_Hiragana_Katakana)<0 || GKS(XK_Henkan)<0)){
knflag=0;
A_YA_flag=0;
}

else if(GKS_Lt1()<0) _1ST[0]=0;
else if(GKS(L2)<0) _1ST[0]=1;
else if(GKS(L3)<0) _1ST[0]=2;
else if(GKS(L4)<0) _1ST[0]=3;
else if(GKS(L5)<0) _1ST[0]=4;

else{
/*if(GKS_(XK_Control_R)>=0){*/
     if(_1ST[1]<0){
     if(GKS_Rt1()<0) _1ST[1]=0;
else if(GKS(R2)<0) _1ST[1]=1;
else if(GKS(R3)<0) _1ST[1]=2;
else if(GKS(R4)<0) _1ST[1]=3;
else if(GKS(R5)<0) _1ST[1]=4;
}
else if(_2ND[1]<0){
     if(GKS_Rt1()<0) _2ND[1]=0;
else if(GKS(R2)<0) _2ND[1]=1;
else if(GKS(R3)<0) _2ND[1]=2;
else if(GKS(R4)<0) _2ND[1]=3;
else if(GKS(R5)<0) _2ND[1]=4;
}
/*}
else{
     if(GKS('l')<0) {boin_b=0+1;bbflag=1;}
else if(GKS('o')<0) {boin_b=1+1;bbflag=1;}
else if(GKS('p')<0) {boin_b=2+1;bbflag=1;}
else if(GKS('@')<0) {boin_b=3+1;bbflag=1;}
else if(GKS('[')<0) {boin_b=4+1;bbflag=1;}
}*/
}

printflag=0;
addflag=0;

/*if(bbflag)    sokuon=0;*/
/*if(LCtrlflag) {shift=0;knflag=0;}*/

/*if(!hana && _3flag && _4flag) {qflag=1;_3flag=_4flag=0;knflag=0;}*/

/*if(!hana){
if(_4flag) {if(chouon) {hana=1;knflag=0;printf("hana=1\n");}}
}
else{
if(_4flag) {hana=0;knflag=0;_4flag=0;printf("hana=0\n");}
}*/
```

```
if(active && chouon==1){
if(!hana){
strcpy(kana,ch);
if(chousoku==1/* || sokuon_*/){
     /*if(chousoku==1) {*/strncat(kana,zu,SZ);/*}*/
/*else if(sokuon_)     {strncat(kana,wa,SZ);}*/
kana[SZ*2]='\0';
i=i_=0;j=j_=0;k=1;l=1;m=0;
printflag=/*2*/1;
boin=/*-1+1*/boin;
}
else{
i=i_=0;j=j_=0;k=0;l=1;m=0;
printflag=1;
boin=boin;
}
}/**if(!hana)**/
else{
strcpy(kana,"b");
i=i_=0;j=j_=0;k=boin_b;l=2;m=0;
printflag=-1;
boin_b=boin_b;
}/**else(!hana)**/
}/**if(active,chouon)**/
else{
if(!A_YA_flag){
if(_1ST[1]>-1){
if(shift) {if(_1ST[0]>-1 && _1ST[0]<5) _1ST[0]+=5;}  /* ha ma ya ra wa */
if(BPha/*HA*/ && _1ST[0]<0) _1ST[0]=5;
strncpy(kana,&kn[knflag][0+SZ*AS*_1ST[0]+SZ*_1ST[1]],SZ);

/*if(bbflag) sokuon=0;*/

if(sokuon/* || sokuon_*/){
kana[SZ*1]='\0';
     if(sokuon)  strncat(kana,zu,SZ);
/*else if(sokuon_) strncat(kana,wa,SZ);*/
kana[SZ*2]='\0';
i=_1ST[0]+1;i_=_1ST[1]+1;j=j_=/*3*/0;k=0;l=0;m=knflag;
printflag=4;
boin=-1+1;
}
else{
kana[SZ*1]='\0';
i=_1ST[0]+1;i_=_1ST[1]+1;j=j_=0;k=0;l=0;m=knflag;
printflag=3;
if(_1ST[0]==9){
     if(_1ST[1]==0) boin=_1ST[1]+1;  /* わ */
else if(_1ST[1]==1) boin=4+1;        /* を */
else if(_1ST[1]==2) boin=2+1;        /* ん */
else                boin=_1ST[1]+1;
}
else                boin=_1ST[1]+1;
}
}/**if(_1ST[1])**/
```

```
}/**if(!A_YA_flag)**/
else if(A_YA_flag==1){              /* a i u e o */
if(_2ND[1]>-1){
if(shift) {if(_1ST[0]>-1 && _1ST[0]<5) _1ST[0]+=5;}  /* ha ma ya ra wa */
if(BPha/*HA*/ && _1ST[0]<0) _1ST[0]=5;
strncpy(kana,&kn[knflag][0+SZ*AS*_1ST[0]+SZ*_1ST[1]],SZ);

kana[SZ*1]='\0';
strncat(kana,&kn[3][0+SZ*AS*0+SZ*_2ND[1]],SZ);
kana[SZ*2]='\0';

/*if(bbflag) sokuon=0;*/

if(sokuon/* || sokuon_*/){
     if(sokuon)  strncat(kana,zu,SZ);
/*else if(sokuon_) strncat(kana,wa,SZ);*/
kana[SZ*3]='\0';
i=_1ST[0]+1;i_=_1ST[1]+1;j=0+1;j_=_2ND[1]+1;k=1;l=0;m=knflag;
printflag=6;
boin=-1+1;
}
else{
i=_1ST[0]+1;i_=_1ST[1]+1;j=0+1;j_=_2ND[1]+1;k=0;l=0;m=knflag;
printflag=5;
boin=_2ND[1]+1;
}
}/**if(_2ND[1])**/
}/**else if(A_YA_flag)**/
else if(A_YA_flag==2){              /* ya yu yo */
if(_2ND[1]>-1){
if(shift) {if(_1ST[0]>-1 && _1ST[0]<5) _1ST[0]+=5;}  /* ha ma ya ra wa */
if(BPha/*HA*/ && _1ST[0]<0) _1ST[0]=5;
strncpy(kana,&kn[knflag][0+SZ*AS*_1ST[0]+SZ*_1ST[1]],SZ);

kana[SZ*1]='\0';
strncat(kana,&kn[3][0+SZ*AS*7+SZ*_2ND[1]],SZ);
kana[SZ*2]='\0';

/*if(bbflag) sokuon=0;*/

if(sokuon/* || sokuon_*/){
     if(sokuon)  strncat(kana,zu,SZ);
/*else if(sokuon_) strncat(kana,wa,SZ);*/
kana[SZ*3]='\0';
i=_1ST[0]+1;i_=_1ST[1]+1;j=0+2;j_=_2ND[1]+1;k=1;l=0;m=knflag;
printflag=8;
boin=-1+1;
}
else{
i=_1ST[0]+1;i_=_1ST[1]+1;j=0+2;j_=_2ND[1]+1;k=0;l=0;m=knflag;
printflag=7;
boin=_2ND[1]+1;
}
}/**if(_2ND[1])**/
}/**else if(A_YA_flag)**/
```

```
}/**else(active,chouon)**/

if(printflag){
/*getkp();*/
if(kp<0) {printf("kp<0 -> No characters processed\n");goto end;}
killflag=1;
#if kanaDATA==3
strcpy(str2,"");
#endif

len=strlen(kana);

/* espk only */
if(chouon<0 && printflag>2){
if(printflag%2==0 && sokuon){       /* boin=-1+1 */
/*len=strlen(kana);*/
kana[len-SZ]='\0';                  /* strips っ */

boin=get_boin(/*_1ST[0]*/_1ST0old,_1ST[1]);     /* redefinition */
/*printf(" boin:%d\n",boin);*/

strcat(kana,"ーっ");
/*nn=0;*/
/*     if(len==SZ*2)  {j=4;j_=3;k=1;}
else if(len==SZ*3)  {k=2;l=1;}*/
}
else{
strcat(kana,"ー");
/*     if(len==SZ*1)  {j=4;j_=3;}
else if(len==SZ*2)  k=2;*/
}

addflag=1;
}/**if(chouon,printflag)**/
else if(chouon>0){
if(!strcmp(kana,"ー")){
strncpy(kana,&kn[0][(boin-1)*SZ],SZ);kana[SZ]='\0';
strcat(kana,"ー");                  /* あー, いー, うー, えー, おー */
nn=0;
addflag=1;
/*i=1;i_=boin;j=4;j_=3;k=0;l=0;m=0;*/
}
else if(!strcmp(kana,"ーっ")){       /* chousoku:1, boin=boin */
strncpy(kana,&kn[0][(boin/*k*/-1)*SZ],SZ);kana[SZ]='\0';
strcat(kana,"ーっ");                /* あーっ, いーっ, うーっ, えーっ, おーっ */
nn=0;
addflag=1;
/*i=1;i_=boin;j=4;j_=3;k=1;l=0;m=0;*/
}
}/**else if(chouon)**/

if(nn && boin){
strcat(kana,"ん");
addflag=1;
/*     if(len==SZ*1) {j=5;j_=3;}
```

```
else if(len==SZ*2)  k=3;
else if(len==SZ*3)  l=2;*/
}

len=strlen(kana);
stmp[0]=kana[len-1-2];stmp[1]=kana[len-1-1];stmp[2]=kana[len-1];stmp[3]='\0';
if(/*!strcmp(stmp,"っ") || */!strcmp(stmp,"ん")){

if(/*len==SZ*3*/0){
/*stmp[0]=kana[len-1-2-SZ];stmp[1]=kana[len-1-1-SZ];stmp[2]=kana[len-1-SZ];stmp[3]='\0';
if(!strcmp(stmp,"ー")) ;else boin=-1+1;*/
}/**if(len,SZ*3)**/
else{
boin=-1+1;
}

if(addflag) printflag=10;
}/**if(!strcmp(),!strcmp())**/

/*js=*/puts_3B(kana);

BitBlt_full();

if(printflag==2 || (!hana && !boin)) active=0;else active=1;
/*if(printflag==1 || printflag==3) sysflag=0;else sysflag=1;*/
/* espk only */
/*if(1) sysflag=1;*/



if(chouon>0){
i=0;i_=boin;j=j_=0;
if(chousoku) k=1;else k=0;
l=1;m=0;n=kp+1;
}/**if(chouon)**/
else{
if(sokuon) k=1;else if(nn) k=2;else k=0;
if(chouon<0) l=1;else l=0;
m=knflag;n=kp+1;
}/**else(chouon)**/

#if (espk_af==0 && kanaDATA==2) || (espk_af==1 && kanaDATA)
printf("%s i:%d i_:%d j:%d j_:%d k:%d l:%d m:%d n:%d\n",kana,i,i_,j,j_,k,l,m,n);
#endif

#if espk_af==1
strcpy(str2,X_PATH);
strcat(str2,"/[");

/*gcvt(i,10,buff);*/
strcpy(buff,itos(i));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(i_,10,buff);*/
```

```
strcpy(buff,itos(i_));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(j,10,buff);*/
strcpy(buff,itos(j));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(j_,10,buff);*/
strcpy(buff,itos(j_));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(k,10,buff);*/
strcpy(buff,itos(k));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(l,10,buff);*/
strcpy(buff,itos(l));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(m,10,buff);*/
strcpy(buff,itos(m));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(n,10,buff);*/
strcpy(buff,itos(n));
strcat(str2,buff);
strcat(str2,"].");

strcat(str2,FT);
#endif

#if 1
if(espk_af>=1 || strcmp(kana,"ん")){
#else
strncpy(stmp,kana,3);stmp[3]='\0';
if(strcmp(stmp,"ん")){
#endif
if(espk_af==0){
strcpy(str,"espeak-ng");

/*strcat(str," -v ja+f3 -a 100 -s 200 -p ");*/
strcat(str," -v ja+");
strcat(str,mf);

strcat(str," -a ");
/*gcvt(BPa,10,buff);*/
strcpy(buff,itos(BPa));
strcat(str,buff);
```

```
strcat(str," -s ");
/*gcvt(BPs,10,buff);*/
strcpy(buff,itos(BPs));
strcat(str,buff);

strcat(str," -p ");

/*gcvt(ff_fc(pitch[kp]),10,buff);*/
strcpy(buff,itos(ff_fc(pitch[kp])));
strcat(str,buff);
strcat(str," ");

strcat(str,kana);
}/**if(espk_af)**/
else if(espk_af==1){
strcpy(str,AUDIO);
strcat(str," -q ");

strcat(str,str2);
}/**else if(espk_af)**/
#if espk_af==2
else{
/*_add2_kana(kana,kp);*/
AUDIO(kana,kp);
/*AUDIO(kana);*/
}/**else(espk_af)**/
#endif

if(espk_af<=1){
if(BPbreak) kill_PID(AUDIO);

strcat(str," &");
system(str);
#if espk_af==0 && kanaDATA==1
printf("%s %d\n",kana,kp);
#endif
#if kanaDATA==3
printf("str:%d str2:%d\n",strlen(str),strlen(str2));
#endif

/*printf("str:%s\n",str);*/
}
}/**if(strcmp(kana,"ん"))**/
#if espk_af==0
else{                                /* kana:ん */
/*i=10;i_=3;j=j_=k=l=m=0;*/
#if kanaDATA
printf("%s i:%d i_:%d j:%d j_:%d k:%d l:%d m:%d n:%d\n",kana,i,i_,j,j_,k,l,m,n);
#endif

         /* ha */
strcpy(str2,X_PATH);
strcat(str2,"/[");

/*gcvt(i,10,buff);*/
```

```
strcpy(buff,itos(i));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(i_,10,buff);*/
strcpy(buff,itos(i_));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(j,10,buff);*/
strcpy(buff,itos(j));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(j_,10,buff);*/
strcpy(buff,itos(j_));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(k,10,buff);*/
strcpy(buff,itos(k));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(l,10,buff);*/
strcpy(buff,itos(l));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(m,10,buff);*/
strcpy(buff,itos(m));
strcat(str2,buff);
strcat(str2,"][");

/*gcvt(n,10,buff);*/
strcpy(buff,itos(n));
strcat(str2,buff);
strcat(str2,"].");

strcat(str2,FT);

/*strcat(str," -v ja+f5 -a 200 -s 200 -p ");
if(1) val=70;else if(0) val=kp*12.5;else val=pitch[kp];*/

strcpy(str,Audio);
strcat(str," -q ");

strcat(str,str2);
strcat(str," &");
system(str);
#if kanaDATA==3
printf("str:%d str2:%d\n",strlen(str),strlen(str2));
#endif
}/**else(strcmp(kana,"ん"))**/
#endif
```

```
end:
knflag=0;
A_YA_flag=0;

end_:
shift=0;
chouon=0;
chousoku=0;
sokuon=0;
nn=0;

_1ST0old=_1ST[0];
if(BPha/*HA*/) _1ST[0]=-1;

_1ST[1]=-1;
_2ND[1]=-1;
}/**if(printflag)**/

/*prtflag=printflag;*/
if(BPi){
if(!printflag) indicator(shift,_1ST[0],chouon,sokuon,nn,knflag,A_YA_flag);
else           indicator(shift,_1ST0old,chouon,sokuon,nn,knflag,A_YA_flag);
}

end_Escape:
#endif

return 1;
}/**if(event.type)**/
else if(event.type==ButtonPress){
XQueryPointer(d,w,&root,&child,&rx,&ry,&wx,&wy,&mask);
if(wy>=YRESO-1-ds*3 && wy<YRESO-1-ds*2 && wx>-1 && wx<XRESO){
BPs=/*0*/1+(min(wx/ds,9)+/*1*/0)*BPd;
if(espk_af==0) printf("s:%d\n",BPs);
}

return 1;
}/**else if(event.type)**/
else if(event.type==Expose){
bitblt(1,0,0,XRESO,YRESO,0,0);
/*indicator(shift,_1ST[0],chouon,sokuon,nn,knflag,A_YA_flag);*/

/*printf("Expose\n");*/

return 1;
}/**else if(event.type)**/
else if(event.type==MotionNotify){
getkp();

if(kp<0){
#if espk_af<=1
if(killflag){
if(BPbreak) kill_PID(AUDIO);
killflag=0;
```

```
}
#else
_add2_nosound();
#endif

/*refill=0;*/
}

if(kp!=oldkp){
/*_add2_inform_pitch(kp);*/
oldkp=kp;
}

return 1;
}/**else if(event.type)**/
#if espk_af==2
else if(event.type==_add2_Press){
getkp();

if(kp>-1) _add2_resound(kp);
/*if(kp>-1) _add2_resound();*/

return 1;
}/**else if(event.type)**/
#endif

return 0;
}/** wndproc **/
#endif
```