

# AUDIBLE COMPUTING --- Masa Kasahara

What would happen if we only rely on our hearing when using a computer? Masa will discuss his project to use Forth to provide an audible interface to a minimal computer, just a motherboard with firmware. He'll describe his thinking and his goals and solicit advice from the group.



# What is Audible Computing?

- My (tentative) Definition: To perform any kind of computing tasks based on audible interface rather than visual or other physical means.



# Why FORTH?

- Theoretically, any languages can be used for this project as long as it is represented in alphanumerical format. – Most of the languages, however, are not designed for audible computing in mind.
- Any characters can also be used, theoretically. – Chinese characters, for example, are too complicated to implement. – Do we have Chinese Morse Code?

# Tower of Hanoi

```
void move(int d, char f, char t) {  
    printf('moving disk %d : %c --> %c', d, f, t);  
}  
  
void hanoi(int h, char f, char t, char r) {  
    if (h > 0) {  
        hanoi(h-1, f, r, t);  
        move (h-1, f, t);  
        hanoi(h-1, r, t, f);  
    }  
}
```



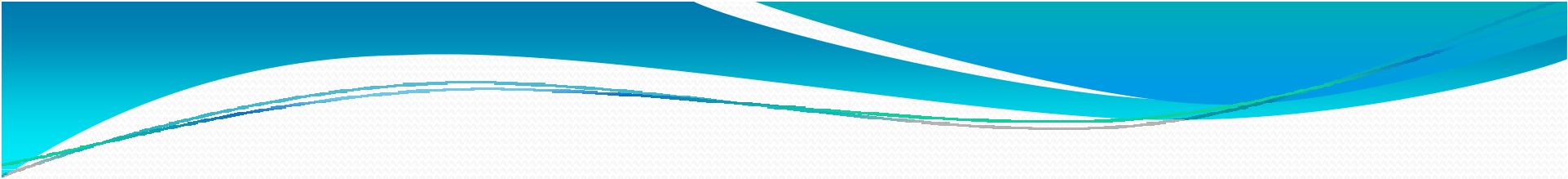
# We can talk about it forever, or actually experiment it.

- It is probably much quicker to experiment it rather than examining hypothetical scenarios because I don't know what is like to program entirely in audible fashion. – I am not an academia. I am more interesting in practicality.
- Now, the question is how to begin.



# Prototype 1:

- Instead of building a FORTH system with Morse Code Interface built-in, how about a terminal program written in FORTH with Morse Code Interface? This terminal program can be connected to another FORTH system. This approach is probably easier since Morse Code Interface is running as an application though this separation may not be apparent in FORTH programming.



# Serial Terminal with Morse Code Interface: What we need?

- Item 1: Find a good FORTH implementation. It should run on a regular PC. The source should be available and well documented. It is also nice if the system runs off of a floppy.
- Item 2: Find a cheap PS/2 mouse to hook up a squeeze key. Eventually, left and right mouse switches will be connected to the contacts of a squeeze key.

# How Morse Code is constructed: Brief Explanation

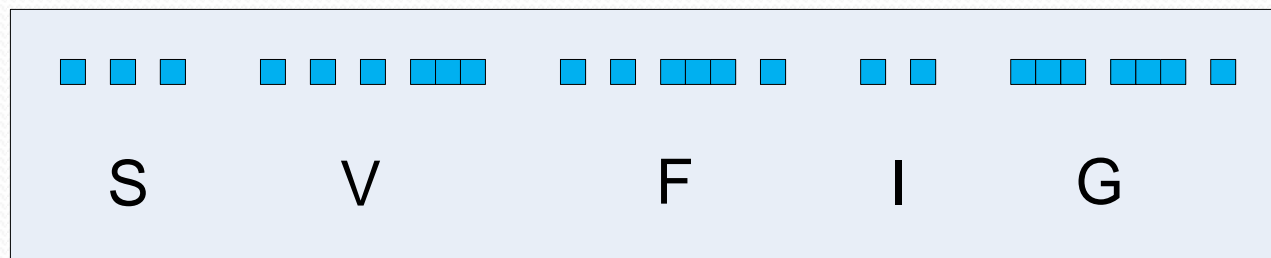
- Timing is everything in Morse Code:

Dot: 1 [Unit length, which changes according to the speed of keying.]

Dash: 3 [Dot length=Unit length]

Character Space: 3

Word Space: 7





# Keying Devices:

- Traditional Straight Key – The program has to determine if the input is Dot or Dash. You can count the switch-on time and compared against the reference, which is not difficult, but more complicated than the following squeeze key.



Photo by K4EQ. Copyright released.

# Keying Devices: (continued)

- Modern Squeeze Key – The separate input for Dash and Dot, which makes programming and keying easier.



K7SRA Key



# Requirements:

- There are two tones for Morse Code: one for incoming, the other for outgoing for clarity. They should be adjustable for personal preferences.
- Any unrecognized characters in input stream will be simply discarded. – No complicated terminal emulations, but a simple teletype emulation, abbreviated as TTY.
- Any unrecognized character in keying stream will be interrupted with a correction symbol for an immediate re-transmission, which makes it easier to use.



# Requirements: (continued)

- A space should be explicitly keyed in. This is not a regular Morse Transmission. It is an interface to a computer. There will be a lot of pauses during programming. It is too complicated to distinguish implied spaces from simple pauses in keying stream.
- Speed should be adjustable for personal preferences. The program should start with the lowest acceptable speed.



# Serial TTY with Morse Code Interface (continued)

- Step 1: Find simple FORTH terminal routine samples.
- Step 2: Implement Morse Code Output routine.
- Step 3: Hook up another FORTH system with a serial cable and experiment it.
- Step 4: Implement Morse Code Input routine.
- Step 5: Experiment.

# Darci USB - Morse Code Computer Access

- <http://www.westest.com/darci/usbindex.html>



Darci USB

Photo: Courtesy of  
Westest Engineering Corporation

# Special Symbols:

The Morse Code Extensions, created by Westest Engineering Corporation, No Copyright

- **Enter \*-\*- Space \*\*-- Back Space ---- Tab -\*--\***
- **Tab Left --\*-\*<sup>\*</sup> Underscore \*\*--\* Page Up ---\*\*-**
- **Page Down ---\*-\* Left Arrow ----\*-**
- **Right Arrow -----\* Up Arrow ----\*\***
- **Down Arrow ----- Escape \*\*-\*<sup>\*</sup> Home \*\*\*\*\*-\***
- **End -\*-\*<sup>\*</sup> Insert \*-\*\*<sup>\*</sup> Delete -\*\*\*<sup>\*</sup>**
- **Start Menu --\*\*\*\*\***

# Special Symbols: (continued)

- **STICKY KEYS**

- **Shift \*\*-\*.** **Alt \*-\*--** **Ctrl -\*-\*.**

- **Windows \*\*-\*--** **Application Key -\*\*\*--**

- **COMMAND CODES**

- **Caps Lock \*\*-\*.\*** **Mouse Mode --\*.\***

- **Number Mode -\*\*\*-** **Repeat Mode \*-\*\*\*.\***

- **Sound On/Off \*-\*.\*** **Code Set ---\*.**





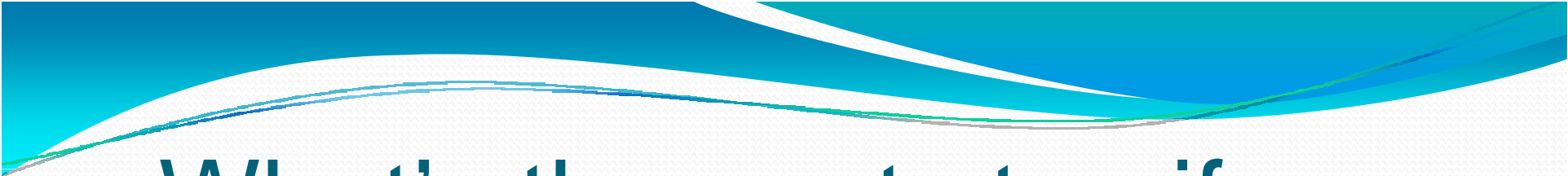
# Evaluation of Prototype 1

- This prototype should give us a reality of Audible Computing, whether the idea works or not. If it does not work out, why?
- I am particularly concerned with the back and forth between an operator and machine, such as:
  - 1) How easy the error correction process is.
  - 2) If errors are found, is break-in possible? This could be an issue for this approach since ordinary FORTH systems are not designed for Morse Code Interface in mind.



# What are potential applications of Prototype 1

- Monitoring of text output devices while engaging other tasks.
- It is probably not effective as an Interface Device for visually impaired since Text-to-speech technologies are dramatically improved over the years.
- By the way, I came across an organization called Sensory Access Foundation in Sunnyvale.



# What's the next step if Prototype 1 is found usable?

- Separation of the implementation dependent routine from the program to make porting easier.
- Implement the Morse Code Interface into a particular FORTH system.
- Since the Interface is directly interacting with the kernel, I tend to think there is an opportunity for improved interactions with operators, possibly with additional features not possible within a terminal program. Now, it's time for Prototype 2.



# Prototype 2:

- Prototype 1 is focused on the feasibility of the concept whereas Prototype 2 is focused on actual implementation of the concept.
- Find Open Firmware for PCs. In worst case, use my SPARCstation 4. Open Boot is a FORTH system.
- The target FORTH system should be modified for Audible Computing, especially in editing (or human interface) area. Preferably, implement a switch to change the mode of operation so that the original interface is kept intact. Or, if a keyboard and/or video interface does not present, it automatically switches into Morse Code mode.



# Disclaimer:

- This is created best to my knowledge at the time of writing. It is, however, not guaranteed to be 100% accurate. I shall not be held liable for either direct or indirect damages by your applying the information contained here. You are at your own.



# Copyright Notice:

- This work is copyrighted by Masa Kasahara.
- You are granted to make copies and share with your colleagues for educational purposes.
- All the images used are either my own, if not specified, or in public domain with sources indicated.
- **Third-Party Trademarks:** Brand and product names are or may be trademarks or registered trademarks of their respective holders. All rights with respect to those trademarks or registered trademarks are reserved by their respective holders.