

White Paper

Speech and Command Recognition

September 21, 2010

Rudan Bettelheim—Freescale

David Steele—Arcturus



Speech and Command Recognition

Voice Controlled Machine Interfaces

The machine interface has come a long way since punch cards and clunky switches. High-resolution graphics, capacitive touch screens, proximity sensors and accelerometers have created a generation of devices that provide a rich interaction experience. For situations where a visual human machine interface is unsuitable or too expensive, voice is a practical alternative. Voice output has been available for some time, but voice control has been a unique challenge even though there are a number of embedded applications that are particularly suited to it:

- Industrial applications and voice HMI
- Home automation and appliance control
- Point-of-sale and automated vending
- Warehouse dispatch and inventory picking
- Order taking and fast serve restaurants
- Health care monitoring and assisted living
- Building control, particularly elevators

The challenge with voice control is overcoming the variance between human voices, which can be more or less of an issue depending on the desired implementation. Cellular phones addressed this by having the user record “voice tags,” each becoming associated to a function, for example “call Bob.” The pre-recorded tag is compared to the spoken command and if the algorithm determines a positive match, the function is executed and Bob is called. This speaker-dependent model is ideal for personal devices and well suited for simple embedded applications, it consumes little CPU resources and does not require large storage for a complex dictionary. The disadvantage is that it is user specific and requires training for each voice tag. This model is therefore impractical for multi-user devices or devices that require a dynamic vocabulary. Consider having to record a new voice tag every time you uploaded a new song into your music catalog.

PC systems, in contrast, have had speech-to-text applications for several years. These tools can be used as full-featured interfaces to the desktop or as plug-in dictation applications. Some PC-based PBX server implementations include interactive voice response (IVR) tools for call routing and queuing. Unlike embedded devices, PC systems have the resources to support large vocabularies and complex acoustic models. These systems are built on speaker-independent models that require no training and work with multiple users. The drawbacks are significant CPU resources, even for a modern PC processor, and mass-storage space for dictionaries worth of data. This has put speech-to-text systems beyond the realm of most embedded devices.

The objective of speech recognition on embedded devices is to provide something better than speaker-dependent voice tags, but not as powerful or versatile as real-time speech-to-text. Lets call this “voice control.” Consider the needs of a fairly simple vending machine; we can understand that every product in the machine would require a name, the names would change as the products change and the machine would need to understand an array of people in order to process each order. Therefore, the requirement would be something like:

- A speaker-independent acoustic model to handle multiple users
- A vocabulary large enough to be able to recognize a broad range of words
- Flexibility to change the recognized words

- The ability to recognize word sequences (commands), not just single words
- Does not need to be “real-time” (i.e. not speech-to-text)

To achieve these requirements, this paper will look at some of the influencing factors that have brought speech recognition to embedded devices and focus specifically on an application package available for the Freescale ColdFire MCF5301x family of processors and as part of Arcturus Voice and Media Middleware. The application package is based on the open source Sphinx project from Carnegie-Mellon University.

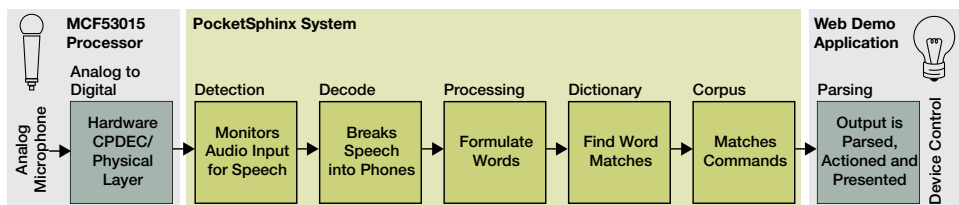
The Confluence of Innovation

Speech recognition was introduced at the 1962 World’s Fair where IBM demonstrated the future of office automation with a device called the “shoebox.” A speech decoder connected to an adding machine capable of recognizing digits 0–9. While this innovation revolutionized nothing, it was noteworthy because of the way IBM used acoustic features to determine the articulatory characteristics found in each spoken number. This foreshadowed the modern method of speaker-independent speech recognition.

By the late 1960s advancements in mathematics brought forward a new concept known as the Hidden Markov Model (HMM). This probabilistic algorithm was used to discover the transitions between known states, in other words, the hidden states. It took an additional 20 years for the combined benefit of acoustic features and HMM algorithms to be realized in speech processing, the result of which created the “mathematic fingerprint” used today.

As compute power increased in the 1980s commercial speech applications emerged from humble beginnings (1000 words at 10 percent accuracy). Around 2000, driven largely by military funding, new projects emerged such as Carnegie-Mellon’s Sphinx project, with the dual objectives of creating open acoustic models as well as applications that can make use of them.

Designing and Implementing Embedded Speech Recognition



Audio Subsystem

The physical interface of a speech processing system is the same as any digital voice system. It requires analog-to-digital conversion by a hardware codec and in the case of systems that are using a microphone input, some pre-amplification may be required. Most speech recognition systems are designed to handle 16-bit PCM audio data, sampled at either 8 or 16 kHz. It’s useful to note that this format makes the interface with the application compatible with telephony vocoders used for VoIP such as G.711 (8 kHz) or G.722 (16 kHz) and thus compatible with telephone infrastructure.

Sampling and Speech Preparation

The speech recognition application will continually sample the audio input adjusting for varying background noise conditions. Since most speech recognition applications do not require real-time speech-to-text conversion, the audio data can be buffered and processed in chunks. This is accomplished by a fixed buffer length or by detecting silence at the end of a phrase, thus signalling the application that the spoken command has been completed and may begin processing.

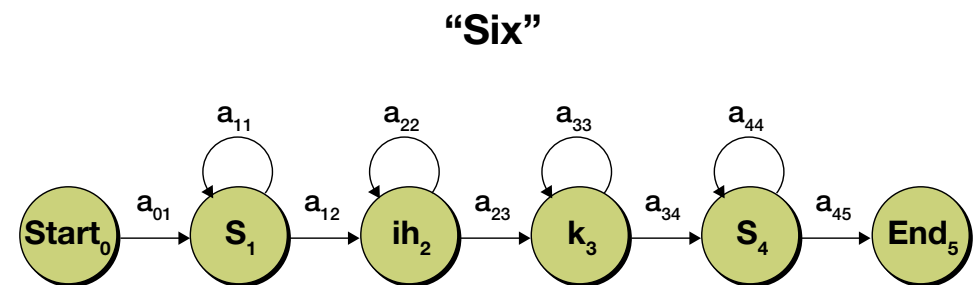
As the processing takes place, the system prepares the incoming audio data and uses an MFCC algorithm to define the acoustic observations or feature vectors of the audio. These include pitch, change in pitch, sound profile and momentary pauses.

The Acoustic Model

The acoustic model is the statistical representation of sounds, which is built up by recording and converting spoken word through a training process. Both speaker-dependent and independent systems require acoustic models to decode speech. With speaker-dependent systems, such as voice tags, the acoustic model is developed by learning the way certain words are pronounced by the user and storing a reference with which to compare them.

Speaker-independent systems overcome the training issue by using an acoustic model that represents a wide cross section of speakers (age, sex and dialect). This cross section is then processed to form word clusters that are then sub-divided into sub-clusters to form patterns and tokens. This resource is stored locally as the acoustic model. Speaker-independent systems are independent by virtue of the large amount of training information already contained in the acoustic model.

In a speaker-independent system, a user's speech is broken down into its phonetic subcomponents and processed by an algorithm to extract a sort of mathematic fingerprint of the sound. This is called an acoustic observation and consists of the feature vectors of the acoustic sound. From there, the HMM algorithm is then used to predict the likelihood of the transition from one feature vector to the next (the hidden states). Ultimately, this chain of processing combined with other likelihood elements, such as the sequential nature of speech, results in the detection of the most probable word.



Where S1, ih2 k3 s4 start and end are the observable states with the hidden states (represented by 'a') weighted by probability. The self-loops account for the variability in pronunciation length,

-Heather Dewey-Hagborg

deweyhagborg.com

It's important to note that the acoustic model is a separable component in the speech processing system and can be thought of (using an over simplification) like a plug-in. This allows the model to be re-trained for various languages, modified to perform better for certain dialects, without affecting the rest of the system. Open acoustic models, (those that are publicly available) such as those included with the Sphinx project, include an ecosystem of tools to tune and create the acoustic model.

The acoustic model breaks down the incoming audio to discover the phonemes, partial phonemes and senones, a senone being the smallest sub-phonetic unit of speech for which hidden state probability can be calculated. A specialized algorithm called a Viterbi path search is used to evaluate the most probable hidden states from within a predefined acoustic model.

The use of phonemes is key, as the English language has as huge vocabulary but has a relatively limited number of phonemes (less than 50). By discovering the phonemes, we can start to search

for phonemes, phoneme clusters and probable word matches. Phoneme clusters, referred to as diphones (two) and triphones (three) can be used to greatly increase the probability of a word match.

In the case of Pocketsphinx, there are two main acoustic models available for North American English: TIDIGITS and WSJ1. The TIDIGITS corpus consists of more than 25 thousand digit sequences spoken by over 300 men, women and children. The data was collected in a studio and originally digitized at 20 kHz, then down-sampled to 8 kHz. This acoustic model is designed for number recognition only and is not generally a good model for natural speech.

In contrast, the WSJ1 acoustic model contains approximately 78,000 training utterances (73 hours of speech), 4,000 of which are the result of spontaneous dictation by journalists with varying degrees of experience in dictation. In this acoustic model, the subjects were recruited to read Wall Street Journal article paragraphs excerpted pseudo randomly. The recordings were made using two microphones and encoded at 16 kHz then down-sampled to produce a second 8 kHz version. This results in a dictionary that understands about 100,000 unique words and, by virtue, is significantly larger than the TIDIGITS model but more suitable for general speech applications.

Command and control applications have various requirements beyond the use of digits only and, as such, the TIDIGITS acoustic model is somewhat limiting. A better implementation choice for most command-based speech recognition systems is to use the generalized acoustic model found in WSJ1.

Language Model and Sentence Corpus

After this stage of processing, a phonetic picture of a word emerges. If we use the previous example "SIX," the output would look something like "Si K s." A matching algorithm compares this output to a language model. The language model is a dictionary of known words that has been pre-processed for phonetic equivalence. For example, the word "SIX" would be represented by two columns

```
SIX      Si K s
```

This makes the pattern matching between the acoustic model and the language model fairly straight forward.

Similarly, a sentence corpus provides a particularly useful tool for command matching. The application can therefore increase the probability of a match by looking at an entire command and not just a single word. In the example provided earlier, the dictionary may contain the following words:

```
VANILLA
```

```
ICE
```

```
CREAM
```

```
PIE
```

The sentence corpus could contain the actual name as well any natural equivalents a user might speak, for example:

```
VANILLA ICE CREAM PIE
```

```
VANILLA PIE
```

```
ICE CREAM PIE
```

```
ICE CREAM PIE VANILLA
```

Applications such as home automation or even drive-thru restaurants support a relatively limited vocabulary. This means that the already narrow focus of the language model, combined with the

increased accuracy obtained through a sentence corpus, can result in an interface equally as reliable as other methods that rely on definitive user input.

The drawback to this model is that while common words will be identified in the language model, some names and words that may be specific to an application or industry may need to be added. A tool can be used to process the plain text to produce the phonetic equivalence and store the entry in a custom “hand” dictionary. For example, the Linux command “syslog,” a concatenation of “system log,” gets added to the “handdict” as SYSLOG S IH Z L AH G.

Implementation of a Command Control System

In Arcturus middleware, the voice control interface is provided as part of an application framework and supports a simple out-of-the-box demo. The demo allows users to log into the device’s Web interface and assign names to speed dials and commands to I/O. The speed dials or I/O can then be commanded through the voice control system, making it possible to quickly create voice-driven applications for intercom systems, building access and in-home monitoring. The demo also provides an interface to the middleware to allow basic configuration and voice response commands to “say IP address” and “say phone number.” This simple interface is a way for equipment installers to check system parameters without the added cost of an LCD screen or the need for a console connection.

At the development level, the voice control interface provides an example parser that post-processes the output of the speech recognition system looking for matches. The process signals “ready” as well as “success” or “failure” for command matches. When successful, the system will execute a predefined action, script, middleware or system command. This fairly simple parser/matcher provides a powerful voice control interface that can be used for a broad range of applications. To increase the extensibility of the system, a tool is pre-integrated which can rebuild the language model on-demand and store it in read/write flash. This means that new functions can be added or the system can be customized for a particular site. The out-of-the-box demo makes use of this framework as an implementation example, and C-code is provided.

The ColdFire Advantage

Most low- to mid-range 32-bit processors do not have sufficient performance for audio (voice) processing. And most low- to mid-range DSPs do not have enough control capability to both control an application and manage a network connection. This generally necessitates the use of both a 32-bit CPU and a DSP for digital voice applications.

Most of Freescale’s ColdFire embedded processors include an enhanced multiply accumulate unit (EMAC), which enables them to process audio and run the central application as well as manage a network connection. This additional processing power can be used to off-load computationally intensive functions such as MFCCs, FFTs or transcoding directly into hardware. Using a ColdFire processor instead of a CPU/DSP combo results in a simpler system with a lower total cost.

In addition to the EMAC, ColdFire processors, in particular the MCF53017, include a rich on-chip peripheral set that is suitable for various audio applications. The MCF53017 MCU provides both dual 10/100 Ethernet and an internal 8 kHz, 16-bit audio codec with speaker and headphone amplifiers, and microphone preamp. This means that all D/A or A/D processing occurs directly inside the processor with no need for an external hardware codec. External peripheral devices, such as keypad scanners, I/O controllers or EEPROMs, can be connected to the host processor via USB, DMA SPI or I²C. Interrupts, GPI/O, UARTS and two smart card interfaces are also available. What’s more, the MCF53017 processor contains two Ethernet controllers and an SDIO controller, which makes it ideal for wired or wireless network applications.

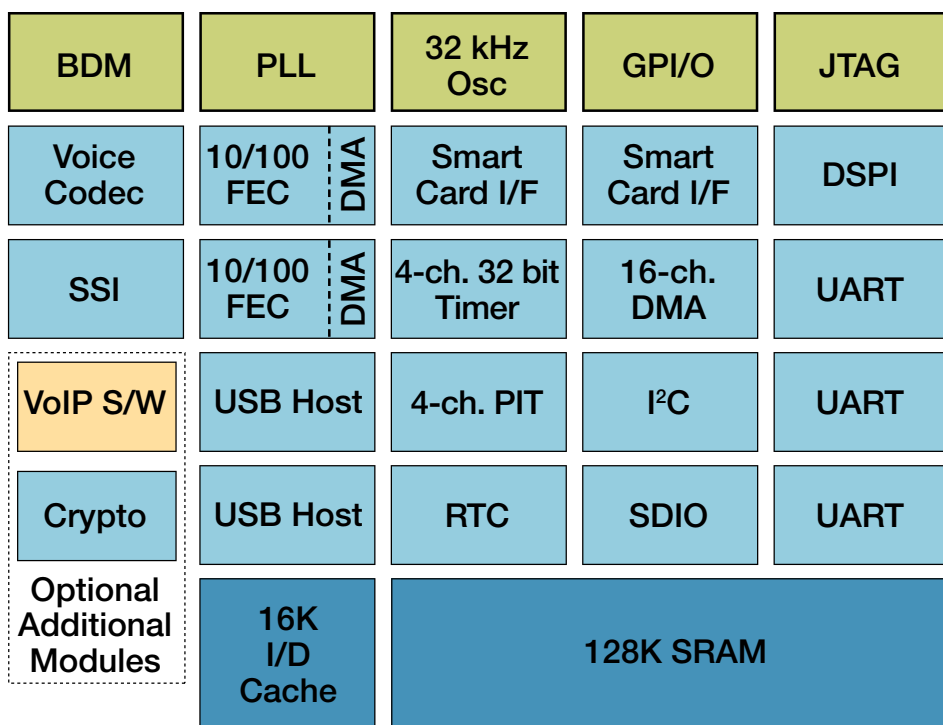


Figure 2: MCF53017 block diagram

System Resources and Optimizations

From a development standpoint, it's important to note that the WSJ1 acoustic model is a single 5 MB file and needs to be loaded into RAM. For devices with memory management hardware, this is generally not such an issue, as the MMU can handle the contiguity. For MMU-less devices, especially in memory constrained systems, planning is required to ensure the allocation can be made contiguously. Depending on the allocator being used, particularly ones that use the power-of-two model, this estimate may be overly conservative and try to allocate sometimes up to twice as much RAM as is required. One counter measure for a large allocation is ensuring the application starts early in the boot sequence to ensure the allocation is serviced.

Areas are being reviewed for optimizations and improvements:

- Parser: Currently, the parser requires exact word and phrase matches. It should be possible to make the parser smarter or more forgiving to overcome natural "ah's or "um" and increase the spoken language understanding.
- Processing: Investigation needs to be done to see the hardware acceleration such as the EMAC and SRAM can be used to off-load any advanced mathematics currently executed as user instructions.

Processing	Percent of Time
Codebook	35.17
HMM	31.62
MFCC	16.46
Senone	16.74

Demo Systems and Additional Information

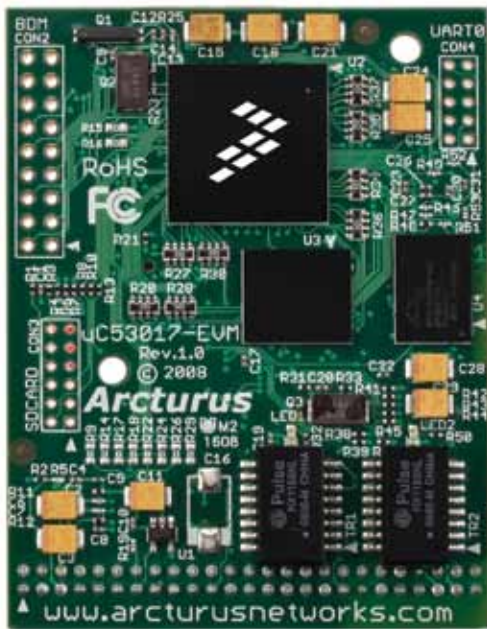
A fully functional demonstration of the voice control system is available as part of the MCF5301x family evaluation kit, the M53015EVB. Detailed information on the evaluation kit is available at: freescale.com/webapp/sps/site/prod_summary.jsp?code=M53015EVB

The Freescale MCF5301x product page also includes a link to a video of the voice control demo: freescale.com/webapp/sps/site/prod_summary.jsp?tid=mcHp&code=MCF5301x



The voice control demo is also available on the VoIP-based digital voice development kit, available from Freescale as M53017KIT and from Arcturus as uC53017-Start Kit, or as a production-ready module, M53017MOD, from Freescale and uCC53017H50-32EE64UVM. Additional information is available on the Arcturus website at:

arcturusnetworks.com/products/uc53017/



If you already have the M53015EVB evaluation kit and would like to evaluate the latest version of the voice control functionality, you can download it at:

arcturusnetworks.com/MCF53015-EVBDEMO.zip

For more information on Arcturus digital voice and speech and command recognition solutions, visit arcturusnetworks.com or e-mail: arcsupport@arcturusnetworks.com

Attributions

Pocketsphinx: A Free Real-Time Continuous Speech Recognition System for Hand-Held Devices

David Huggins-Daines, Mohit Kumar, Arthur Chan, Alan W Blac, Mosur Ravishankar and Alex I Rudnicky, 2006; www.cs.cmu.edu/~awb/papers/ICASSP2006/0100185.pdf

Speech at CMU, Carnegie Mellon University, www.speech.cs.cmu.edu/ and the Sphinx team, www.speech.cs.cmu.edu/people.html

Additional Reading

Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition, Daniel Jurafsky, James H. Martin, Prentice Hall, 2009.

Speaker Independent Connected Speech Recognition, Fifth Generation Computer Corporation, www.fifthgen.com/speaker-independent-connected-s-r.htm

How to Reach Us:

Home Page:

freescale.com

i.MX Portfolio Information:

freescale.com/iMX

e-mail:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
1-800-521-6274
480-768-2130
support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447
303-675-2140
Fax: 303-675 2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright license granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

For more information, visit freescale.com

Freescale, the Freescale logo and ColdFire are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners. © 2011 Freescale Semiconductor, Inc.

Document Number: SPCMRECWP / REV 0

