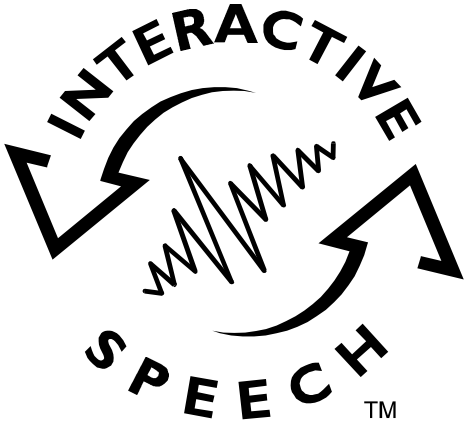
Voice Dialer
Data Book



SENSORY

Copyright

Trademarks

IMPORTANT NOTICES

Sensory reserves the right to make changes to or to discontinue any product or service identified in this publication at any time without notice in order to improve design and supply the best possible product. Sensory does not assume responsibility for use of any circuitry other than circuitry entirely embodied in a Sensory product. Information contained herein is provided gratuitously and without liability to any user. Reasonable efforts have been made to verify the accuracy of this information but no guarantee whatsoever is given as to the accuracy or as to its applicability to particular uses. Applications described in this data book are for illustrative purposes only, and Sensory makes no warranties or representations that the Voice Dialer series of products will be suitable for such applications. In every instance, it must be the responsibility of the user to determine the suitability of the products for each application. Sensory products are not authorized for use as critical components in life support devices or systems.

Sensory conveys no license or title, either expressed or implied, under any patent, copyright, or mask work right to the Voice Dialer series of products, and Sensory makes balance between recognition and synthesis no warranties or representations that the Voice Dialer series of products are free from patent, copyright, or mask work right infringement, unless otherwise specified. Nothing contained herein shall be construed as a recommendation to use any product in violation of existing patents or other rights of third parties. The sale of any Sensory product is subject to all Sensory Terms and Conditions of Sales and Sales Policies.

Technical Support

For product support and questions:

Marketing Communications
521 East Weddell Drive
Sunnyvale, CA 94089-2164
Tel 408-744-9000

Technical Support
521 East Weddell Drive
Sunnyvale, CA 94089-2164
Tel 408-744-9000
techsupport@sensoryinc.com

Table of Contents

List of Figures

List of Tables

This page is intentionally blank.

Voice Dialer

Introduction

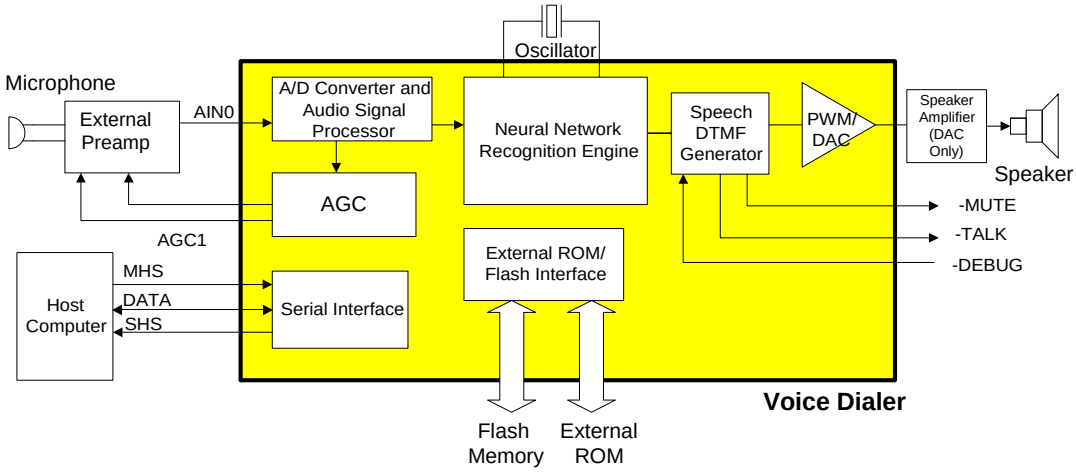
Sensory's Voice Dialer brings advanced speech recognition technology to a variety of telephony consumer products. Designed for embedded systems with low computing power requirements, the Voice Dialer is quickly and easily integrated with product host processors. The Voice Dialer adds functionality, ease-of-use, and leading-edge technology features to products with a minimum of external memory and hardware. This makes Sensory's Voice Dialer ideal for cordless phones, hand-held dialers, and practically any portable consumer electronics product using voice-activated telephony technology.

This data book describes the end-user voice recognition features, training approach, and command set, as well as the design, interface, and protocol parameters you need to implement Voice Dialer technology.

Feature Overview

The Voice Dialer is a single-chip CMOS device. The Voice Dialer uses sophisticated speech recognition technology to map spoken names to phone numbers. Using easy-to-learn, intuitive techniques, end users train the Voice Dialer, which performs speaker-dependent (SD) speech recognition, speech prompting, and DTMF (Dual Tone Multi-Frequency) synthesis. The Voice Dialer is available in IC, IC Module, and Evaluation Board formats.

Figure 1 - Block Diagram



As shown in Figure 1 - Block Diagram, the Voice Dialer system includes ROM as well as host interfaces, analog-to-digital signal conversion, analog control signals, fully-automated speech prompting (neural network recognition engine and speech generator), and PWM or DAC output. Among the Voice Dialer's capabilities:

- A telephone directory with up to 60 names (with a 2 megabit memory; up to 30 names with a 1 megabit memory) with the following for each entry:
  - Speech recognition template
  - Digital recording of template name for audio confirmation
  - Phone number storage (up to 4 numbers per name)
  - An attribute byte to focus directory searches
- Speaker-dependent access to any name in the directory
- Multiple directory support for more than one user per Voice Dialer
- DTMF generation for responsive, voice-actuated dialing
- User-friendly speech prompting
- Language localization and custom prompt capabilities
- A full command set to handle telephone directories (add, modify, delete)
- A simple serial interface

Sensory's Voice Dialer is designed to operate as a slave chip, receiving and processing commands from a host controller or Master CPU (MCPU) and returning status information and data. The Voice Dialer has private access to its own control program (external memory or internal masked ROM), to an optional language/extended speech ROM, and to a non-volatile external read/write Flash memory. Either 1 or 2 megabits of flash memory are supported – the size reflects

application-specific requirements for the phone directory. Communication with the MCPU is through a 3-wire serial bus.

The Voice Dialer recognizes a rich command set allowing the telephone application programmer to implement complex voice recognition functions with a minimum of MCPU overhead. This allows the application software to focus on providing an intuitive and efficient user interface.

Implementation Overview

Implementing Voice Dialer technology involves writing application control software for a host controller (MCPU) and physically embedding a Voice Dialer IC into your product.

What follows is an overview of Voice Dialer concepts and capabilities, its command set and serial interface, and its hardware requirements and features. Subsequent sections take up these topics in detail.

The Voice Dialer maintains a dialing directory with the following information for each entry:

- A speaker-dependent speech template – a digitally constructed synthesis of a spoken name
- A voice recording of the name – the system plays the recording for confirmation
- Four 0-30 digit strings – the telephone numbers associated with names
- An attribute byte – to identify the kind or type of entry during searches

Users can access all data associated with a name entry either by voice recognition or sequentially. Users can also organize multiple directories or word groups – these capabilities make use of the attribute byte. The attribute byte is combined with a search mask during recognition and directory functions. This makes it possible to search through phone numbers assigned by multiple users and searching by categories (names of business contacts versus names of friends, for example).

All operations are controlled by the MCPU through the Master/Slave bus. The Voice Dialer can be programmed to provide dialing information to the MCPU or to directly generate DTMF signals. The MCPU can also request single-digit DTMF dialing from the Voice Dialer. An extensive, easy-to-implement command set provides a complete interface for the application control software. Your program can incorporate speech from a long list of pre-recorded prompts to creatively guide the end user through the user interface. Commands are conveyed in data packets via the 3-wire serial interface that connects the host processor to the Voice Dialer.

The Voice Dialer requires minimal external circuitry (as shown in Figure 8 - Voice Dialer IC Schematic). The following components are required unless otherwise noted:

- Microphone
- Preamplifier
- Flash Memory
- Oscillator
- Speaker
- Speaker Amplifier (optional)
- External ROM for customized speech or foreign language (optional)

A full line of Sensory Voice Dialer products is available to help you develop the application:

- *The Voice Dialer IC* - The Voice Dialer integrated circuit without the external components listed above.
- *The Voice Dialer Module* - A small pre-configured PCB for prototype development. The Voice Dialer Module includes the Voice Dialer IC and all of the required external components except for a microphone and speaker.
- *The Voice Dialer Evaluation Kit* - A fully configured Voice Dialer Evaluation Board that incorporates the Voice Dialer IC Module, speaker, microphone, battery pack and more. The kit also includes Sensory's graphical VoiceHost software to evaluate, demonstrate, and train the Voice Dialer.

Voice Dialer Concepts and Functional Capabilities

The following sections describe concepts and capabilities that are useful in creating the application control program that runs on the host computer (MCPU). In the descriptions that follow, “user” refers to the end user of the application control program. The following topics are covered:

- Voice Dialer Entry Structure
  - Speech Templates
  - Voice Recordings
  - Phone Numbers
  - Attribute Bytes
- Masks
- Current Pointer
- Standard Words and Phrases
- Custom Words and Phrases
- Development Parameters

Voice Dialer Directory Entry Structure

The Voice Dialer creates and stores entries in its directory. Entries have the following elements:

- A speaker-dependent speech template of the name
- A voice recording of the name
- Up to four phone numbers of up to 30 digits for each entry
- An attribute byte

These are described below.

Speaker-Dependent Speech Templates

Within the context of the product application, a user must program the Voice Dialer to train each name in the directory, and then enter the corresponding phone number(s). During this training process, speaker-dependent speech templates are created and then stored for comparison during the recognition process.

Training is simple. It consists of speaking a name and then repeating it for confirmation. Each time the name is spoken a template is created. The two templates thus created during the training process must closely match (the speech patterns should be similar). If the two templates match sufficiently, then a third template composed of the average of the first two templates is created. This averaged template is then stored in external flash memory and compared during the recognition process.

During the voice recognition process, a fresh template of the spoken word to be recognized is produced. This new template is then compared to the stored templates to determine which name was spoken.

Voice Recordings

The Voice Dialer creates a voice recording of each stored name. This voice recording can be played back before or after dialing to confirm that the correct entry was selected by the recognition routines. If users wish to change the voice recording associated with a particular number, they can use a software command to re-record names without first having to delete and recreate entire entries.

The voice recording is generated during the training process, after the two speaker-dependent templates have been recorded. This recording will vary in length depending on whether the speech compression function is enabled. If compression is disabled, the maximum length of the recording is 1.5 seconds in length. However, if 4-bit to 2-bit compression is enabled, the recording can last up to 2.0 seconds.

Phone Numbers
After the Voice Dialer creates the speaker-dependent (average) template and generates the voice recording, users should specify at least one phone number (up to 30 digits, including pauses for credit card dialing) to complete the entry creation process. If no phone number is stored, then a 'dummy' number, or a zero-length phone number, must be saved instead.

Attribute Bytes
When a phone number is added to a newly created entry, an attribute byte is associated with the voice template. Attribute bytes allow users to distinguish between different groups or kinds of entries.

Consider two users, Troy and Jennifer, who share the same 60-entry Voice Dialer. When Troy trains the Voice Dialer to recognize his name, a speech template, voice recording, and an attribute byte – 00000100b("b" indicates binary) – is stored in an entry. This attribute byte is used to mark a user directory, that is, a special kind of entry. (Of course, to create a valid entry, Troy also must enter a dummy telephone number.) Jennifer's name occupies another entry, but also stores the attribute byte – 00000100 – along with her speech template and voice recording (and a dummy number). These first two entries are of a special kind – they do not store useful phone numbers, but instead two categories of users.

Now there are 58 entries remaining. As Troy and Jennifer fill up the directory with names and telephone numbers, the Voice Dialer creates user-specific attribute bytes to organize names and telephone numbers into two categories: Troy's entries and Jennifer's entries. For example, Troy's entries may share an attribute byte of 00000001b, while Jennifer's entries would each have an attribute byte of 00000010b. Other bits in the attribute byte make possible further categorization. For example, suppose Jennifer trains the Voice Dialer to recognize the words "business" and "friends" to organize her telephone numbers. Jennifer's attribute byte for business names might be 00010000 and for friends 00001000. Within each business or friend entry, there are up to four phone numbers (for example, home, work, pager, and cell phone numbers).

The attribute byte will normally have one bit position set to one, while the rest remain at 0. This allows for differentiating up to eight different kinds of templates. Although the attribute byte is used to categorize the names of the separate phone numbers, it is not used to select one of the four stored phone numbers of the current entry.

This method of dividing trained names into categories requires no extra memory or buffering because the Voice Dialer dynamically allocates memory rather than using fixed blocks. Consider an application that supports two different users. Rather than dividing the memory in half, it can be allocated according to actual usage. The user with more numbers to call can train templates as needed, while the other user fills up

the remainder. This avoids leaving empty memory slots in one user's directory, while the second directory is needlessly and inconveniently restricted to fewer slots.

> *Unlike entries that store telephone numbers, directory name entries -- "Troy," "Jennifer" -- store dummy telephone numbers. When writing the application control program, you can use the dummy number slot in a directory name entry to store useful information, such as the attribute for the directory. See the Set Attribute and Mask in the command summary.*

Masks

During searches, the MCPU control program generates a mask. The mask is logically 'and'ed against each attribute byte. Masks support flexible applications with multiple directories or word lists. In fact, masks help such applications perform more responsively by reducing search times. During recognition and directory functions, a mask is constructed to select and/or exclude words in specific groups. In our previous example, Troy trains all numbers with an attribute of (00000001), while Jennifer trains all numbers with an attribute byte of (00000010). When Troy's voice is recognized by Voice Dialer, a mask byte of 00000001 is generated and searches only numbers trained by Troy.

> *During the training process, the mask is used to determine which existing word groups should be included when testing for similar words. The mask also has an effect on how other directory commands such as "Increment Current Pointer," "Decrement Current pointer," and "Query Lexicon Status" will function. For more information, see the command summary.*

Current Pointer

The Current Pointer points to the current entry of a directory and is used to select and operate on entries within a directory. The entry selected by the Current Pointer is referred to as the Current Entry. Many of the Voice Dialer commands operate on the Current Entry.

Standard Words and Phrases

The Voice Dialer has 32K bytes of general-purpose Standard English words and phrases onboard. Two word lists, a Prompt List and a Speech list, have been developed for telephony applications. Each list has its own set of indices.

The Prompt List contains speech that is accessed directly by the chip during the training and recognition process. Speech in this list is also accessed to repeat phone number digits as well as specific error messages (such as "Please talk louder"),

thereby providing user feedback during training and voice recognition. Words and phrases in this list cannot be accessed via the Say Prompt command.

The Speech List contains speech that can be accessed via the Say Prompt Command. These are standard prompts useful in creating the user interface for applications. Examples include prompting to enter or delete numbers, to review operations, and to play error messages (such as “memory full” or “memory empty”) at the application level.

Custom Words and Phrases

The application can replace the onboard standard word list with a customized word list for English or foreign languages via an external ROM chip. This external ROM can contain either 32K or 64K of speech synthesis. The only limitation on the external synthesis is that it must be divided into two lists -- the Prompt and Speech lists described above. The Prompt List in the external ROM must contain the same number of indices as that of the onboard Prompt List, and the synthesis at each index must have a meaning (in the new language) identical to that of the onboard Prompt List (in standard English). This discipline is required because the Prompt List indices reflect functionality that is hard-coded into the Voice Dialer. For example, sentence 0 of the Prompt List Table (“Say Name” -- see below) could be replaced with a custom voice synthesis equivalent in a different language, or with an extended prompt such as “Please say the name of the person to be added to the directory.”

Since the Speech List is accessed solely by the application control code, there are no requirements to either the number of indices, or the semantic meaning of any index. The developer is free to create a Speech List of any desired synthesis required by the application – limited only by the available memory. For more information on creating a custom Speech List, contact Sensory.

> *A rule of thumb: about 800 bytes of memory per word are required for good quality synthesis, and about 400 bytes per word for acceptable quality synthesis. This rule of thumb varies by language.*

Development Parameters

The development parameters control the range and scope of the Voice Dialer application. The application can select settings that anticipate the proximity of users to the microphone, the kind of output (DAC or PWM,) to select and so on. These parameters are set using the Store Parameter Command (see this command in the Command Set Overview p. 18). The table and text below summarize the choices.

Table 1 - Development Parameters

Index	Parameter	Default	Range
00h	Training Level	Level0	00 = Level0 (least stringent, lowest rejections, Most recognition errors) 01 = Level1 02 = Level2 03 = Level3
01h	Environment	Far	00 = Far 01 = Near
02h	Recognition Level	Level0	00 = Level0 (least stringent, lowest rejections, Most recognition errors) 01 = Level1 02 = Level2 03 = Level3
03h	Auto-retry	No	00 = No 01 = Yes
04h-0Fh	RESERVED	N.A.	N.A.
10h	Speech Output Select	DAC	01h = DAC only 02h = PWM only 03h = Both DAC & PWM
11h	DTMF Output Select	DAC	01h = DAC only 02h = PWM only 03h = Both DAC & PWM
12h	Pause time (x 10ms)	64h (1 sec.)	01h – 7Fh
13h	Record Compression	Yes	00 = No compression 01 = 4 to 2 bit compression
20h-3Fh	RESERVED	N.A.	N.A.
40h–5Fh	Developer defined	FFh	00h – FFh

Functional Description

00h – Training Level Selects one of four different sets of training parameters.

01h – Environment Allows the user to configure the Voice for near or far applications. A near application is one in which the microphone is always positioned within 1-2 inches of the user's mouth.

02h – Recognition Level Selects one of four different sets of recognition parameters.

03h – Auto-Retry Indicates whether the Voice Dialer will automatically handle retries when the recognition code is uncertain about a match. (The uncertainty arises when two name are too close to distinguish.) Otherwise, the application code handles this situation. In the case of Auto-Retry = YES, the Command 04 (Recognize Word) automatically prompts the user to repeat the spoken name or command to resolve the uncertainty. In the Auto-Retry = NO mode, the Voice Dialer returns a Command Response of 0Ah (Recognition Uncertain) to the host. The host must then determine the proper course of action (for example, displaying "Recognition Uncertain" on an LCD).

04h to 0Fh – RESERVED. Do not use.

10h, 11h - Speech Output Select & DTMF Output Select Parameters These parameters allow the user to select the analog output device to be used with speech synthesis and DTMF dialing. A value of 01h directs the output to the DAC and a value of 02h directs the output to the PWM. It is legal to select both the DAC and PWM simultaneously by setting these parameters to 03h. The default value for both of these parameters is 01h (DAC output).

12h - Pause Time Parameter This parameters specifies the length of a pause used when a pause function is executed. This value is in 10ms increments and allows for a range of 10ms to 2.5 seconds. The default pause time is 1 second (64h).

13h – Recording Compression Mode This parameter controls the compression of recorded voice during training. If compression is disabled, then the maximum recording length is approximately 1.5 seconds. If compression is enabled, then the maximum recording length is approximately 2 seconds, but the recording quality will be slightly diminished. This parameter can be changed at any time. For example, in a given system, some recordings may be made with compression on and others with compression off.

The Voice Dialer Command Set

Command Set Overview

This section describes the set of Voice Dialer commands and command responses. The commands are sent to the Voice Dialer by the MCPU via the serial interface (see p. 33).

Table 2 - Command Set

Command	Parameters	Data returned	Description	Class
00h	~	~	No Operation	6
01h	~	Version String	Get Version String	6
02h	Prompt, Tries	~	Train A Name	1
03h	Prompt, Tries	~	Recognize Word	2
04h	~	~	Say Current Name	3
05h	N	~	Say Current Number	3
06h	N, Digits	~	Add Phone Number	3
07h	N	Dialing Digits	Return Current Number	3
08h	~	~	Clear Current Pointer	3
09h	~	~	Increment Current Pointer	3
0Ah	~	~	Decrement Current Pointer	3
0Bh	~	~	Save Current Pointer	3
0Ch	~	~	Restore Current Pointer	3
0Dh	~	~	Swap Current Pointer	3
0Eh	~	~	Delete Current Entry	3
0Fh	0x55	~	Delete All Stored Names	3
10h	~	~	Rerecord Name Recording	3
11h	Mask, Attribute	~	Set Mask and Attribute	3
12h	~	Attribute, Status	Get Entry Status	3
13h	~	Capacity, Free, Matching	Query Lexicon Status	3
14h	Prompt #	~	Say A Prompt	4
15h	Digits	~	Say Number String	4
16h	~	~	Power Down	5
17h	Index, Parameter	~	Store Parameter	5
18h	Index	Parameter	Fetch Parameter	5
19h	CP	~	Set Current Pointer	6
1Ah	~	CP	Get Current Pointer	6
1Bh	Digit	~	Dial A Single Digit	7
1Ch	Digits	~	Dial Number String	7
1Dh	N	~	Dial Current Number	7

Class definitions:

1 – Training
2 – Recognition
3 – Directory function
4 – Synthesis
5 – Control function
6 – Debug
7 – DTMF

Command Responses

Table 3 - Command Responses

Response	Class	Description
00h	All	Command successfully executed to completion
01h	1,2	Time out
02h	1,2	Too long
03h	1,2	Too noisy
04h	1,2	Too soft
05h	1,2	Too loud
06h	1,2	Too soon
07h	2	No match found
08h	1	Too similar
09h	1	Not repeatable
0Ah	2	Recognition uncertain
10h	2,3	Memory empty
11h	1	Memory full
12h	3	Invalid pointer (No matching Attributes)
13h	3	No matching Attributes
14h	3	Current Pointer wraparound
20h	All	Unknown command
21h	All	Invalid parameter (value, digit, or index)
FFh	1,2,4	Command interrupted

Functional Description

What follows is a functional description of each command. The invalid parameter response (21h) is always a possible command response, even when not explicitly cited in the descriptions that follow.

Command – 00h (No Operation)
Arguments: ~
Returns: ~
Responses: 00, 21

This command returns a response code of 00h (Success). It is provided primarily for testing the communications interface.

Command 01h - (Get Version)
Arguments: ~
Returns: Version String (56h, 44h, 53h, 02h, 00h)
Responses: 00, 21

This command returns a five-byte sequence indicating the Sensory product model number and software version number. The first three bytes will always be 56h, 44h, and 53h (ASCII 'VDS') for the Voice Dialer System. The last two bytes are, respectively, the major and minor version numbers in binary format.

Command 02h - (Train A Name)	
Arguments:	**Prompt, Tries**
Returns:	**~**
Responses:	**00, 01, 02, 03, 04, 05, 06, 08, 09, 11, 21, FF**

This command allocates directory space and launches a training session for new entries. The training session generates a speaker-dependent speech template and a 2-second voice recording. This command requires two parameters: an index indicating which spoken prompt is to be used (See Table 4 – Training Prompt Codes) and the number of retries allowed. This command may be aborted by the MCPU.

Table 4 – Training Prompt Codes

Prompt #	Spoken Prompt Description - Training
0	"Say name" (Train entry name)
1	"Say directory" (Train directory name)
2	"Say location 1" (Train first location name)
3	"Say location 2" (Train second location name)
4	"Say location 3" (Train third location name)
5	"Say location 4" (Train fourth location name)

The user is prompted using the prompt index supplied, then asked again to "Repeat xxx to confirm," and then prompted one last time to "Repeat xxx to record." The new word is tested for similarity with existing words selected by the current mask. When the entry is saved to permanent memory, the current attribute is also saved.

After the Voice Dialer saves the entry, the Current Pointer points to the newly created entry, but the MCPU must issue an Add Phone Number command to complete the creation process. Other commands, such as the Say Prompt command may be used before the Add Phone Number command, but any command that changes the Current Pointer will cause this new entry to be destroyed. These commands include: Increment, Decrement, Restore, Swap, and Set Current Pointer.

If the training process is unsuccessful after the specified number of retries, the command response will reflect the last attempt (ie., the reported failure will be that of the final try) and returned to the MCPU. If the trained name is too similar to an existing entry name, then the Current Pointer points to the entry that caused the error, but no spoken error is output to the speaker.

Table 5 - Training Response Codes

Response Code	Description	Current Pointer
00	Success	Index of new entry
08	Too similar	Index of entry which is too similar
--	Any other error	Invalid

Command 03h - (Recognize Word)	
Arguments:	**Prompt, Tries**
Returns:	**~**
Responses:	**00, 01, 02, 03, 04, 05, 06, 07, 0A, 10, 21, FF**

This command prompts the user for a name, creates a temporary word template and compares it against stored templates for a match. Specific word sets may be selected using Command 11h (Set Mask and Attribute.) If recognition is successful, the Current Pointer will be set to the index of the entry that matches the word. If the recognition is unsuccessful after the specified number of retries, the Current Pointer will be invalid and the command response will reflect the last recognition attempt and returned to the MCPU. This command requires two parameters: a prompt number to be used, and the number of retries allowed. This command may be aborted by the MCPU.

Table 6 – Recognition Prompt Codes

Prompt #	Spoken Prompt Description – Recognize
0	“Which name?” (Recognize entry name)
1	“Which directory?” (Recognize directory name)
2	“Which location?” (Recognize location name)

Table 7 – Recognition Responses

Response Code	Description	Current Pointer
00	Success	Index of new entry
0A	Recognition Uncertain	Best matching entry (Use Command 0Dh – Swap Current Pointer to select 2nd best matching entry)
--	Any other error	Invalid

Command 04h - (Say Current Name)	
Arguments:	**~**
Returns:	**~**
Responses:	**00, 12, 21, FF**

This command plays the recorded name of the entry associated with the Current Pointer. If the Current Pointer is not valid, the name is not played and the Voice Dialer returns a Command Response of 12h (Invalid Pointer). There are no parameters required for this command. This command may be aborted by the MCPU.

Command 05h - (Say Current Number)	
Arguments:	**N**
Returns:	**~**
Responses:	**00, 12, 21, FF**

This command speaks one of the four phone numbers of the entry associated with the Current Pointer. If the Current Pointer is not valid, the numbers are not played and the Voice Dialer returns a Command Response of 12h (Invalid Pointer). This command requires a parameter from 0-3 to select which phone number is to be spoken. Only the two least significant bits of the parameter are significant. This command may be aborted by the MCPU.

Command 06h - (Add Phone Number)	
Arguments:	**N, Digits**
Returns:	**~**
Responses:	**00, 12, 21**

This command adds or replaces one of the phone numbers in the entry associated with the Current Pointer. If the Current Pointer is set to a new entry, then a new template is created and stored. Command 02h, (Train A Name) must be followed by this command prior to executing any command that changes the Current Pointer. If a phone number is not required for a given word then a null string, dummy number, or an index number must be stored instead. This command requires a parameter from 0-3 to indicate the number to be set or replaced and a string of 0 to 30 dialing digits (see Table 9 - Dialing Digit). Only the two least significant bits of the N parameter are significant.

> *Note: A null string (0 digit phone number) has the effect of deleting an existing phone number.*

Command 07h - (Return Current Number)	
Arguments:	**N**
Returns:	**Digits**
Responses:	**00, 12, 21**

This command returns one of the four phone numbers in the entry associated with the Current Pointer. The data is returned in the format used in Command 06h (Add Phone Number). If the Current Pointer is not valid, a number is not returned; instead, the Voice Dialer returns a Command Response of 12h (Invalid Pointer). This command requires a parameter from 0-3 to select which number to return. Only the two least significant bits of the N parameter are significant. For digits, see Table 9 - Dialing Digit.

Command 08h - (Clear Current Pointer)	
Arguments:	**~**
Returns:	**~**
Responses:	**00, 10, 13, 21**

This command resets the Current Pointer to the first matching entry based on the current mask; it should be used before Commands 09h (Increment Current Pointer) or 10h (Decrement Current Pointer) so that directory entries may be reviewed from a known position. If no entries match the current mask, then this command returns a Command Response of 13h (No Matching Attributes). This command requires no parameters.

Command 09h - (Increment Current Pointer)	
Arguments:	**~**
Returns:	**~**
Responses:	**00, 10, 14, 21**

This command causes the Voice Dialer to increment (move forward) the Current Pointer to the next matching directory entry based on the mask. This command can be used in routines that browse the directory. If the Current Pointer passes the last matching entry, it will wrap around to the first matching entry. In this case, a command response of 14h (current pointer wraparound) is returned.

> **Increment Current Pointer** *and* **Decrement Current Pointer** *are not necessarily linear functions – when used in conjunction with a mask byte they will skip over names that the mask does not match.*

Command 0Ah - (Decrement Current Pointer)	
Arguments:	**~**
Returns:	**~**
Responses:	**00, 10, 14, 21**

This command causes the Voice Dialer to decrement (move backward) the Current Pointer to the previous matching directory entry based on the mask. This command can be used in routines that browse the directory. . . If the Current Pointer passes the first matching entry, it will wrap around to the last matching entry. In this case, a command response of 14h (current pointer wraparound) is returned.

Command 0Bh - (Save Current Pointer)	
Arguments:	**~**
Returns:	**~**
Responses:	**00, 21**

This command saves the Current Pointer in a temporary memory location. The saved Current Pointer value can be restored by Command 0Ch (Restore Current Pointer). This command requires no parameters.

> *Note that the saved Current Pointer is lost if power is removed from the Voice Dialer.*

Command 0Ch - (Restore Current Pointer)	
Arguments:	**~**
Returns:	**~**
Responses:	**00, 12, 21**

This command restores the Current Pointer to its last saved value – the value saved by a Command 0Bh (Save Current Pointer). Note that if directory entry associated with the Current Pointer has been deleted, then the restored Current Pointer will no longer point to a valid directory entry. There are no parameters required for this command.

Command 0Dh - (Swap Current Pointer)
Arguments: ~
Returns: ~
Responses: 00, 21

If, after a recognition attempt is made, there are two possible candidates in the stored speaker-dependent templates that could be the correct entry, the "Recognition Uncertain" command response is returned. In this case, the Current Pointer points to the "first" matching entry. The "Swap Current Pointer" command allows the MCPU to switch the Current Pointer to point to the "second" matching entry. Issuing this command a second time, will switch the Current Pointer back to the "first" matching entry. This allows the MCPU to take appropriate action to resolve which entry is the correct one to use. There are no parameters required for this command.

> *Note that this command is only valid if the "AUTO-RETRY" parameter is disabled (set to "NO").*

Command 0Eh - (Delete Current Entry)
Arguments: ~
Returns: ~
Responses: 00, 12, 21

This command deletes the name, phone numbers, and attribute byte of the entry that is associated with the Current Pointer. If the Current Pointer is not valid, then this command does not delete an entry and instead returns a Command Response of 12h (Invalid Pointer). There are no parameters required for this command.

Command 0Fh - (Delete All Stored Names)
Arguments: 0x55
Returns: ~
Responses: 00, 21

This command deletes all stored names and the Voice Dialer returns to the initial state. This command requires a one-byte parameter of the value 055h.

Command 10h - (Re-record Name Recording)
Arguments: ~
Returns: ~
Responses: 00, 12, 21

This command is used to re-record a name in the directory associated with the Current Pointer. If the Current Pointer is not valid, this command will not re-record a name and instead returns a Command Response of 12h (Invalid Pointer). This command may be aborted by the MCPU. This command will *not* generate any prompts prior to recording.

Command 11h - (Set Mask and Attribute)
Arguments: Mask, Attribute
Returns: ~
Responses: 00, 21

This command is used to set the internal variables mask and attribute when using multiple word sets. Mask and attribute have power up defaults of FFh and 01h, respectively. This allows a single directory of words to be managed without concern for this command. See the sections on Attribute Bytes (p. 13) and Masks (p. 14).

Command 12h - (Get Entry Status)
Arguments: ~
Returns: Attribute, Number Status
Responses: 00, 12, 21

This command returns the attribute and phone number status of the entry associated with the Current Pointer. The first byte sent is the attribute byte of the entry and the next byte indicates how many and which of the four phone numbers are used. However, if the Current Pointer is not valid, this command does not return an entry status and instead returns a Command Response of 12h (Invalid Pointer). There are no parameters required for this command.

Table 8 - Status Byte Definition

Bit	Description
0	Phone number 0 is used (non-null)
1	Phone number 1 is used (non-null)
2	Phone number 2 is used (non-null)
3	Phone number 3 is used (non-null)
4-7	Undefined

Command 13h - (Query Lexicon Status)	
Arguments:	**~**
Returns:	**Capacity, Free, Entries**
Responses:	**00, 21**

This command is used to return information about the lexicon. Three bytes are returned: Total word Capacity, Number of Free slots available, and Number of directory Entries selected by the current mask byte.

> *For Example: Assuming a 2Mbit flash part, the capacity would be 60 total names. If User1 has 30 names trained and User2 has 15 names trained, then the number of free slots available would be 60 – (30 + 15) = 15 free slots. If the current mask is set to select User1, than the entries trained by that user would be 30. In this example, the returned string would be: 60, 15, 30.*

Command 14h - (Say A Prompt)	
Arguments:	**Prompt**
Returns:	**~**
Responses:	**00, 12, 21**

This command plays a predefined voice prompt from the onboard ROM or external Extended Speech ROM. A prompt index is required. Speech prompts are spoken from the Speech List. See the Speech List (p.41). This command may be aborted by the MCPU.

Command 15h - (Say Number String)	
Arguments:	**Digits**
Returns:	**~**
Responses:	**00, 12, 21**

This command is used to speak a number string passed as a command parameter. The required parameter is a string of 1 to 30 dialing digits. This command may be aborted by the MCPU. See Dialing Digits (p-32).

Command 16h - (Power Down)	
Arguments:	**~**
Returns:	**~**
Responses:	**00, 21**

This command causes the Voice Dialer to enter a low power sleep state. This command returns a "successful" response just before entering low power mode.

> *Note that the only method for exiting this low power state is a pulse on the Voice Dialer reset pin. The Voice Dialer cannot be woken up by a serial command.*

Command 17h - (Store Parameter)	
Arguments:	**Index, Parameter**
Returns:	**~**
Responses:	**00, 21**

This command stores a new value for a single parameter in the flash parameter block. The required parameters are an index and value. The flash parameter block provides a number of user configurable options, as well as 32 bytes of general purpose storage. See the section on Development Parameters (p.16).

Command 18h - (Fetch Parameter)	
Arguments:	**Index**
Returns:	**Parameter**
Responses:	**00, 21**

This command returns a single parameter value from the flash parameter block. This command requires an index and returns a value. The flash parameter block provides a number of user configurable options, as well as 32 bytes of general purpose storage. See the section on Development Parameters (p.16).

Command 19h - (Set Current Pointer)	
Arguments:	**CP**
Returns:	**~**
Responses:	**00, 21**

This command sets the internal Current Pointer (CP) to a specific value. The Current Pointer is a binary number representing the slot where a template is internally stored (in the directory). This command requires a new Current Pointer value as a parameter. For more information, see Current Pointer (p. 14).

> *Note that the* **Set Current Pointer** *and* **Get Current Pointer** *commands are intended as a diagnostic tool and are not required for normal operation. Physical and logical relationships are not guaranteed to remain constant. Instead, please use Decrement, Increment, and Clear Current Pointer.*

Command 1Ah - (Get Current Pointer)	
Arguments:	**~**
Returns:	**CP**
Responses:	**00, 21**

This command returns the internal Current Pointer and requires no parameters.

Command 1Bh - (Dial A Single Digit)	
Arguments:	**Digit**
Returns:	**~**
Responses:	**00, 21, FF**

This command dials a single digit using DTMF. This command requires a dialing digit parameter and can be aborted by the MCPU See Dialing Digits (p-32).

Command 1Ch - (Dial Number String)	
Arguments:	**Digits**
Returns:	**~**
Responses:	**00, 21, FF**

This command dials a 1 to 30 digit phone number and requires a string of 1 to 30 dialing digits. This command can be aborted by the MCPU See Dialing Digits (p-32).

Command 1Dh - (Dial Current Number)	
Arguments:	**N**
Returns:	**~**
Responses:	**00, 12, FF**

This command dials the phone number in the entry associated with the Current Pointer using DTMF. (The Current Pointer must be valid.) This command requires a parameter from 0-3 to select which number associated with the entry is to be dialed. This command can be aborted by the MCPU.

Dialing Digits

Table 9 - Dialing Digit Codes indicates which values to use when using Commands 06h (Add Phone Number), 07 (Return Current Number), 15h (say Number String), 1Bh (Dial a Single Digit), and 1C (Dial Number String). Dialing the 'Pause' character results in a short pause between the DTMF tones. The last three digit codes are currently undefined and do not result in DTMF tones, but may be used by the application program to store any other data.

Table 9 - Dialing Digit Codes

Hex digit	Digit represented	Spoken
00h	"0"	"0"
01h	"1"	"1"
02h	"2"	"2"
03h	"3"	"3"
04h	"4"	"4"
05h	"5"	"5"
06h	"6"	"6"
07h	"7"	"7"
08h	"8"	"8"
09h	"9"	"9"
0Ah	"*"	"Star"
0Bh	"#"	"Pound"
0Ch	Fixed length pause	"Pause"
0Dh	Undefined	Undefined
0Eh	Undefined	Undefined
0Fh	Undefined	Undefined

Serial Interface

Communications

Data communication and control are accomplished using a 3 wire synchronous serial interface. Since the serial interface lines are open-collector (open-drain), pull-up resistors should be attached to all signals. If application requirements call for the Voice Dialer to be powered down while the rest of the system is still powered up, take care to ensure that the external pull-up resistors are also powered down. Doing so will prevent current leakage into the Voice Dialer I/O ports. The following table describes the 3 lines used for data and handshaking between the Voice Dialer and the MCPU. In the table and the timing diagrams, VDR refers to the Voice Dialer IC.

Table 10 - Data Transfer Signal Description

Signal	Pin	Description	Use: VDR=>MCPU	Use: MCPU=>VDR
DATA	24	Bi-directional Data line	Data valid when SHS = 0	Data valid when MHS = 0
MHS	39	Master Handshake (MCPU => VDR)	0 = Data bit valid	0 = Data bit accepted
SHS	25	Slave Handshake (VDR => MCPU)	0 = Data bit accepted	0 = Data bit valid

Data is transferred one bit at a time with full handshaking as described below.

1. When the MCPU has data to transmit to the VDR, the MCPU sets DATA to the data value, verifies that SHS (Slave Handshake) is in the *high* state, then sets MHS (Master Handshake) to the *low* state to request a transfer.

2. The VDR senses the *low* state of MHS and reads DATA, which then sets SHS to the *low* state to acknowledge the DATA.

3. The MCPU senses the *low* state of SHS, and sets MHS to the *high* state to indicate that DATA is no longer valid, and at the same time sets DATA high (releasing it).

4. The VDR then sets SHS to the *high* state to indicate that the cycle is complete. Both devices are now ready to transfer the next data bit.

The Voice Dialer remains busy (SHS = low state) after receiving the final bit of a command packet, until after that command has been completed and the Voice Dialer is ready to send a response. During this time, a time-consuming command can be interrupted by the MCPU with a low pulse on the MSH line. When the Voice Dialer has data to transmit to the MCPU, it follows the same procedure by setting SHS to the low state. (See Figures 2, 3, and 4.) The protocol is completely symmetrical. The first processor to set its HS signal to the low state is the transmitter; the other processor is the receiver.

Data is always transferred in 8 bit bytes, with the most significant bit transferred first.

> *Note: By convention, the slave (Voice Dialer) never transmits data unless requested to by the master (MCPU), thus avoiding the possibility of a collision (both processors setting their HS signals low simultaneously).*

Figure 2 - Data Transfer – (Command to Response)

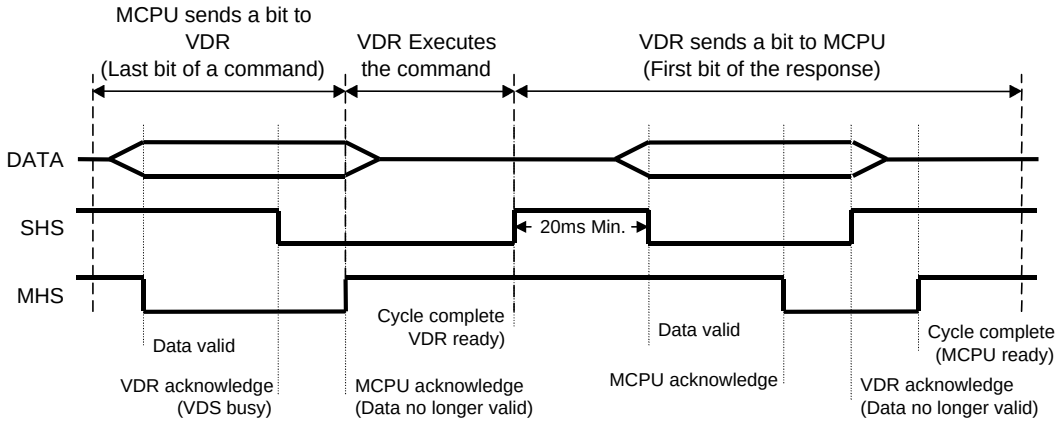


Figure 3 - Interrupt Timing Diagram

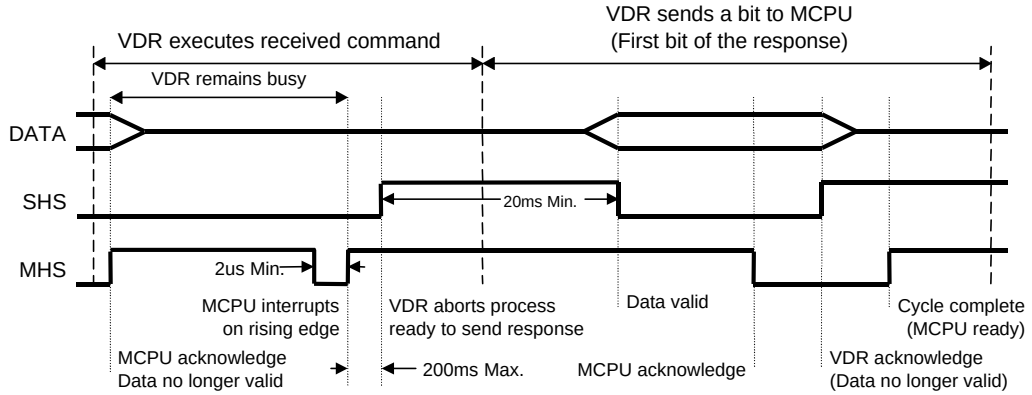
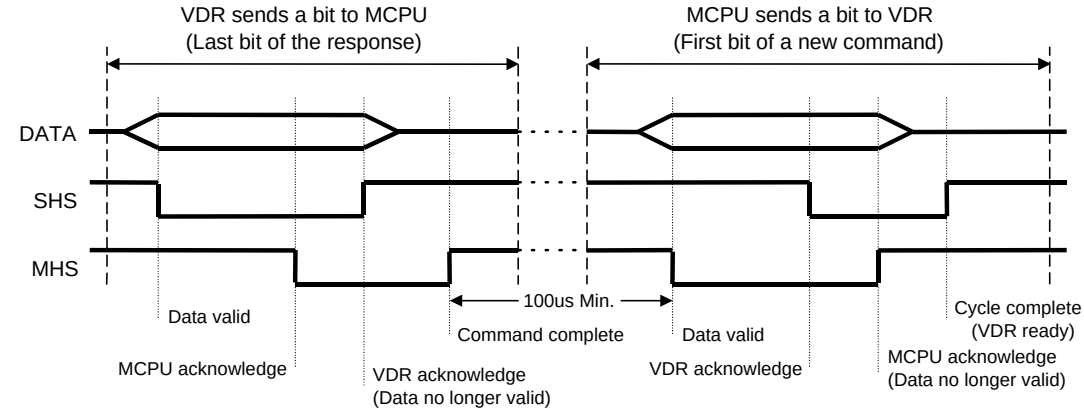


Figure 4 - Data Transfer (Response to Next Command)



Data Packets

Overview

All data is transmitted in 8-bit bytes, and all messages are sent in packets. The packet format conveys error checking and byte synchronization information. A packet always starts with a “sync field” of FFh (8 or more 1 bits) followed by a length byte, then one or more data bytes, then a checksum byte. The length byte specifies the number of bytes to follow including the checksum. The checksum is the 8-bit additive, modulo-256 sum of all the data bytes and the length byte added together. For example the data 01, 02, 03, 04 would be sent as follows:

Table 11 - Sample Data Packet

Byte	Value	Notes
0	FFh	Sync Field, 8 or more consecutive 1 bits
1	05h	Packet length, count of bytes to follow
2	01h	First data byte
3	02h	Second data byte
4	03h	Third data byte
5	04h	Forth data byte
6	0Fh	Checksum, (5+1+2+3+4)

Implementation

Packet/byte synchronization is accomplished as follows:

1. The receiver shifts in bits until the accumulated byte value is equal to FFh (all bits are one).

2. The receiver then continues shifting in data until the most significant bit resets to zero. This indicates that the currently accumulated byte is a valid length byte and the start of a packet.

For synchronization to work properly, the receiver must first initialize its shift register to all zeroes. Once a valid packet is found the receiver accepts (length) bytes and performs the checksum calculation (ascertains that the sum of all the data bytes plus the length byte is equal to the checksum byte). Note that modulo-256 arithmetic is used; the carry is discarded during the checksum calculation.

It is worth recalling that all communication is initiated by the MCPU; the Voice Dialer never sends data unless requested by the MCPU. Accordingly, all commands are in one of the following formats:

- A request from the MCPU to the Voice Dialer. A request consists of a command byte and possible parameters or data.
- A response from the Voice Dialer to the MCPU. A response consists of a status byte and possible data.

After a command is issued to the Voice Dialer, the MCPU must wait until the VDS is ready (SHS = high) before receiving the command response. The MCPU can interrupt a command by pulsing the MHS line to the low state. This causes the Voice Dialer to abort the task and return appropriate status.

External Component Specifications

This section includes a brief description and schematic of the external components needed to fully characterize a Voice Dialer IC. You can acquire these components and then use the schematic to build the circuit. Or Sensory, Inc. can supply you with everything you need:

- *The Voice Dialer Module* A pre-configured PCB for prototype development. The Voice Dialer Module has all of the external components listed in this section and shown in the schematic, except for a microphone and speaker.
- *The Voice Dialer Evaluation Kit* A fully configured Voice Dialer Evaluation Board that incorporates the Voice Dialer Module, speaker, microphone, battery pack and more. The kit also includes Sensory's graphical VoiceHost software to evaluate, demonstrate, and train the Voice Dialer.

For more information on these products, contact Sensory.

Microphone

The microphone is used for speech inputs. It is an inexpensive omni-directional electret model with a minimum sensitivity of -60dB. *Included in the Voice Dialer Evaluation Kit.*

Preamplifier

The preamplifier amplifies and filters the microphone signal to a level suitable for the Voice Dialer. It is either a 2-bit AGC or non-AGC. See the preamplifier schematic. *Included in the Voice Dialer Module and the Voice Dialer Evaluation Kit.*

Flash Memory

The Voice Dialer uses non-volatile memory to store speech templates and phone numbers. It is either a SST29EE010 (1 Mbit), Atmel AT29EE010 (1 Mbit), or SST29EE020 (2 Mbit). *Included in the Voice Dialer Module and the Voice Dialer Evaluation Kit.*

Oscillator

The oscillator provides a high frequency clock (14.32 MHz) for the Voice Dialer. It must be a crystal ceramic resonator or LC circuit. *Included in the Voice Dialer Module and the Voice Dialer Evaluation Kit.*

Speaker

A 32 ohm speaker may be connected directly to the PWM output pins. For better audio, an amplifier and speaker may be connected to the DAC output. *An amplifier and speaker are included in the Voice Dialer Evaluation Kit.*

External ROM (Optional)

The external ROM is optional. It is required for customized speech or non-Standard English language applications. For more information, contact Sensory. *A location for external ROM is included in the Voice Dialer Module and the Voice Dialer Evaluation Kit.*

User Interface

Prompt List

While running training and recognition processes, the Voice Dialer has direct and exclusive access to speech in the Prompt List. This list is also accessed for routines that repeat phone digits. The Prompt List is actually a set a pointers to speech elements listed in the table below.

> *The spoken prompts in the Prompt List can be duplicated in the Speech list with little memory penalty.*

Table 12 - Prompt List

Prompt #	Spoken Prompt Description
0	“Say name”
1	“Say directory”
2	“Say location 1”
3	“Say location 2”
4	“Say location 3”
5	“Say location 4”
6	“Repeat name”
7	“Repeat directory”
8	“Repeat location 1”
9	“Repeat location 2”
10	“Repeat location 3”
11	“Repeat location 4”
12	“Repeat name to record”
13	“Repeat directory to record”
14	“Repeat location 1 to record”
15	“Repeat location 2 to record”
16	“Repeat location 3 to record”
17	“Repeat location 4 to record”
18	“Which name?”
19	“Which directory?”
20	“Which location?”
21	“Which name?” (for retries)
22	“Which directory?” (for retries)
24	“Which location?” (for retries)
25	“0” (flat intonation)
26	“1”
27	“2”
28	“3”
29	“4”
30	“5”
31	“6”
32	“7”
33	“8”

34	“9”
35	“0” (falling intonation)
36	“1”
37	“2”
38	“3”
39	“4”
40	“5”
41	“6”
42	“7”
43	“8”
44	“9”
45	“Star”
46	“Pound”
47	“Pause”
48	<beep> (additional DTMF digit)
49	<beep> (additional DTMF digit)
50	<beep> (additional DTMF digit)
51	(500 millisecond pause) (used for pauses when speaking phone digits)
52	“Louder, please”
53	“Softer, please”
54	“Spoke too soon”
55	“Error”
56	“No match found”
57	<beep>

Speech List

The speech list contains standard prompts you need to create the application’s user interface. You access the speech list via the Say Prompt Command. Phrases can be substituted or added to this list based on memory availability.

Table 13 – Speech List

Prompt #	Spoken Prompt Description
1	“1”
2	“2”
3	“3”
4	“4”
5	“5”
6	“6”
7	“7”
8	“8”
9	“9”
10	“0”
11	“A directory”
12	“A phone number”
13	“Again”

14	“All”
15	“Deleted”
16	“Directory”
17	“Empty”
18	“Enter”
19	“Entry”
20	“Error”
21	“Followed by”
22	“For”
23	“Full”
24	“Location”
25	“Louder”
26	“Memory”
27	“Memory location”
28	“Name”
29	“Names”
30	“Next”
31	“No match found”
32	“Number”
33	“On the keypad”
34	“Phone number”
35	“Please”
36	“Pound”
37	“Press”
38	“Previous”
39	“Repeat”
40	“Say”
41	“Similar to”
42	“Softer”
43	“Spoke too soon”
44	“Star”
45	“Store”
46	“To”
47	“To add”
48	“To confirm”
49	“To delete”
50	“To dial”
51	“To exit”
52	“To modify”
53	“To record”
54	“To review”
55	“Too noisy”
56	“Which”
57	<beep>
58	<beep-beep>
59	(0 millisecond pause)
60	(20 millisecond pause)
61	(40 millisecond pause)
62	(50 millisecond pause)

63	(75 millisecond pause)
64	(100 millisecond pause)
65	(160 millisecond pause)
66	(320 millisecond pause)
67	(1/2 second pause)
68	(1 second pause)
69	(2 second pause)
70	(4 second pause)
71	“Memory full”
72	“Memory empty”
73	“To record location names, press... ”
74	“To record directory names, press... ”
75	“To add a directory, press... ”
76	“To add an entry, press... ”
77	“On the keypad, enter phone number followed by... ”
78	“On the keypad, enter number followed by... ”
79	“To store, press... ”
80	“To exit, press... ”
81	“To dial, press... ”
82	“To delete an entry, press... ”
83	“To delete a phone number, press... ”
84	“To delete a directory, press... ”
85	“To delete all, press... ”
86	“To delete, press... ”
87	“To confirm, press... ”
88	“Entry deleted”
89	“Directory deleted”
90	“Phone number deleted”
91	“To modify an entry, press... ”
92	“To modify a phone number, press... ”
93	“To record again, press... ”
94	“To review an entry, press... ”
95	“To review a phone number, press... ”
96	“To review a directory, press... ”
97	“For next entry, press... ”
98	“For previous entry, press... ”
99	“For next phone number, press... ”
100	“For previous phone number, press... ”
101	“To repeat entry, press... ”
102	“To repeat phone number, press... ”

Voice Training/Recognition Flow Charts

Figure 5 -Training Flowchart (page 1 of 2)

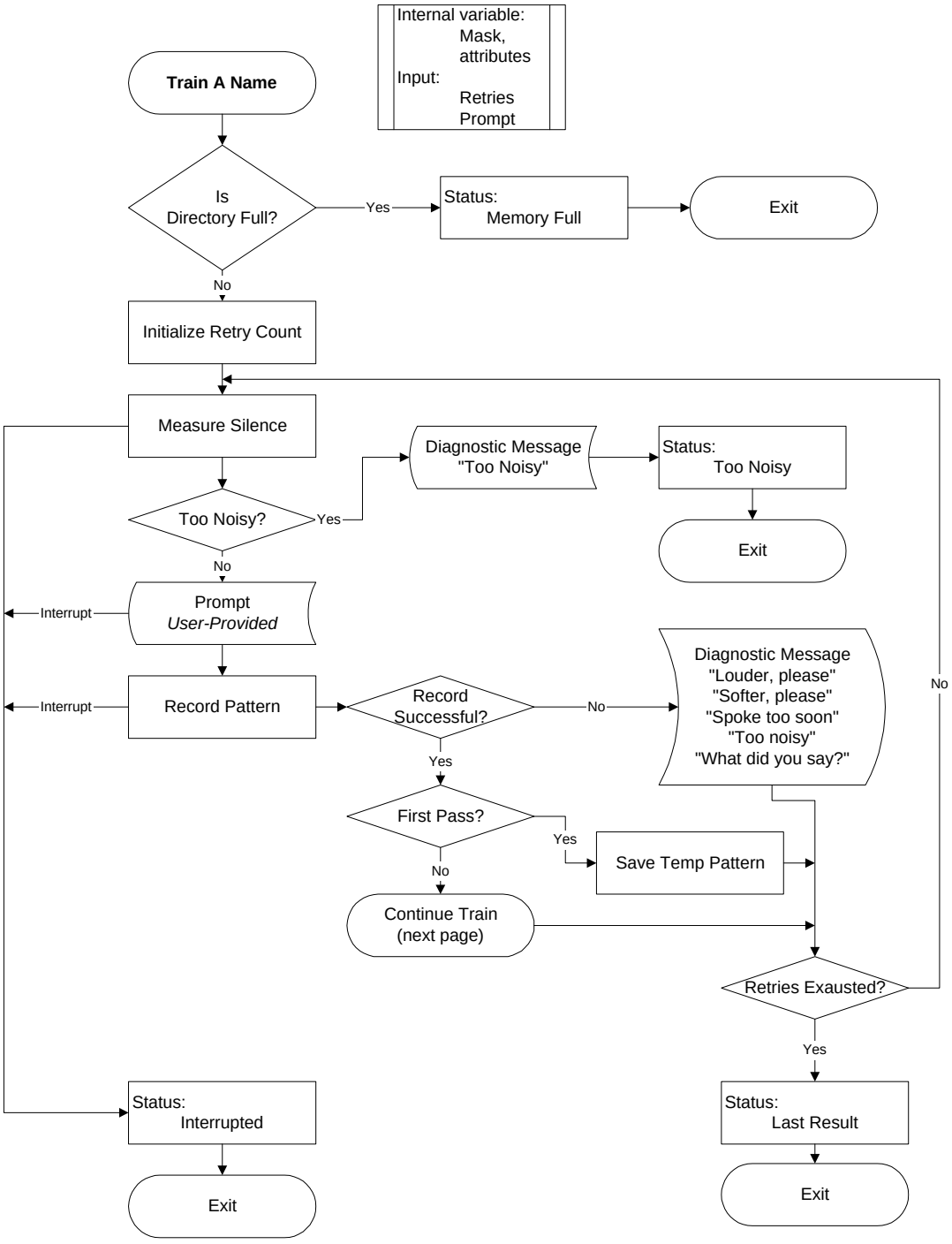


Figure 6 - Training Flowchart (Page 2 of 2)

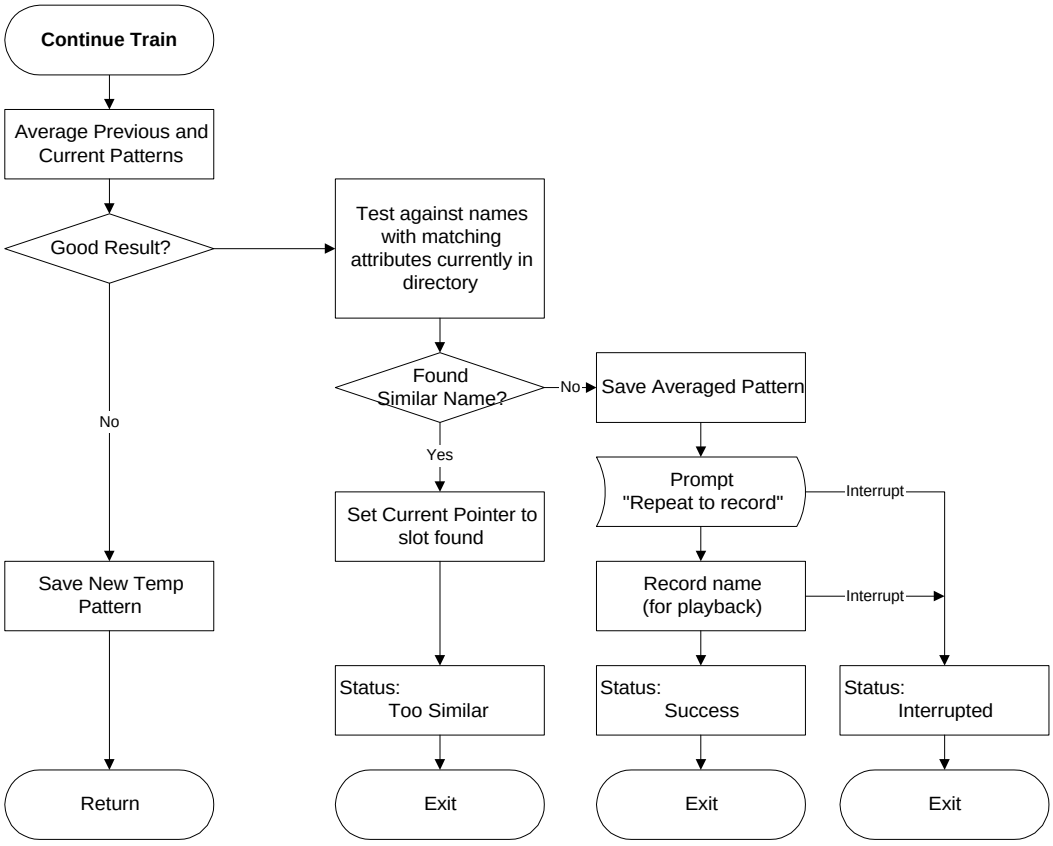
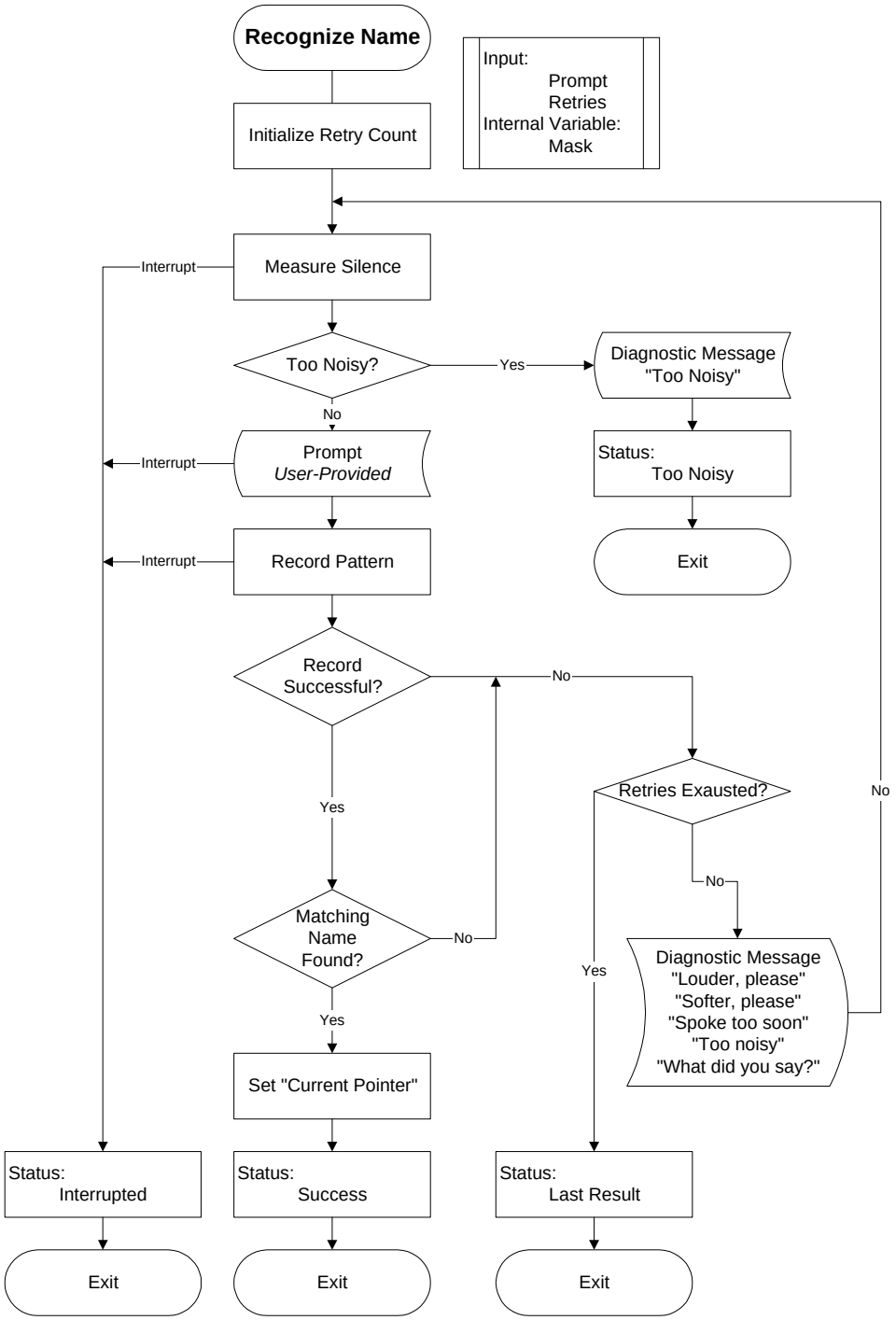


Figure 7 - Recognition Flowchart (Page 1 of 1)



Voice Dialer IC Specifications

IC Pin Descriptions

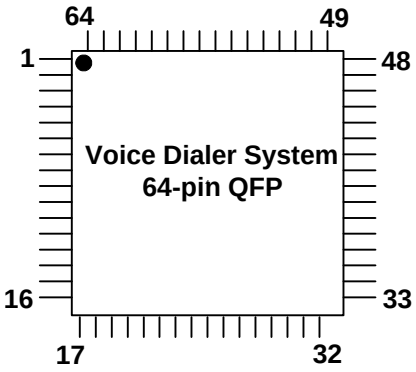


Table 14 - Pin Identification for Voice Dialer IC

Name	QFP Pin	Description	I/O
GND	9, 22 41, 56	Digital Ground, CPU core (pins 1 and 33) and I/O (pins 18and 52)	-
VDD	23	Digital Supply Voltage (core)	-
AGND	52	Analog Ground. For noise reasons, analog and digital grounds should connect together only at the VDS	-
AVDD	55	Analog Voltage. For noise reasons, keep this supply independent of digital circuitry.	-
VDD	10, 40	Digital Supply Voltage (I/O line)	-
-RESET	21	Reset	I
SH	49	Sample and Hold. Connect a 470 pF capacitor from here to AGND.	I
XO1	19	Oscillator 1 output (high frequency)	O
XI1	20	Oscillator 1 input	I
AIN0	51	Analog In, low gain. (range AGND to AVDD/2.)	I
AIN1	50	Analog In, hi gain (8X input amplitude of AIN0, same range)	I
AGC0	33	Audio pre-amplifier Automatic Gain Control line 0	O
AGC1	32	Audio pre-amplifier Automatic Gain Control line 1	O
PWM0	53	Pulse Width Modulator Output0	O
-TE1/PWM1	54	Test Mode *or* Pulse Width Modulator Output1 (multiplexed)	I *or* O
DACOUT	48	Analog Output (unbuffered).	O
-MUTE	30	Audio mute signal. Active low during DTMF output	O
-TALK	31	Audio talk signal. Active low during speech synthesis	O
A[15:0]	1-8, 11-18	External ROM/Flash Memory Address Bus	O
A16	36	External ROM/Flash Memory Address line A16 ROM A15	O
A17	35	External Flash Memory Address line A17	O

D[7:0]	57-64	External ROM/Flash Data Bus	I/O
-RDC	42	External Code Read Strobe	O
-RDD	44	External Data Read Strobe	O
-WRC	43	External Code Write Strobe	O
-WRD	45	External Data Write Strobe	O
MHS	39	Master Handshake. Driven by host.	I
SHS	25	Slave Handshake. Driven by Voice Dialer.	O
DATA	24	Serial Data between Master and Slave. Bi-directional	I/O
DEBUG	34	Enable Debug Diagnostic speech	I
-XMH	46	Default/Custom Speech Select	I
-XML	47	Unused. Must be tied high	I
	26-29, 37-38	Unused	-

Absolute Maximum Ratings

- Minimum voltage on any pin V_{ss}-0.6V
- Maximum voltage on any pin V_{dd}+0.6V
- Operating temperature (T_O) -20°C to +70°C
- Soldering temperature 260°C for 10 sec
- Maximum voltage 7.5V
- Power dissipation 1W
- Storage Temperature -65°C to +150°C
- Minimum Operating Voltage 3.5V
- Maximum Operating Voltage 5.0V

> ***WARNING**: Stressing the Voice Dialer beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability*

DC Characteristics

(T_O = -20°C to +70°C, Vdd = 5V)

SYMBOL	PARAMETER	MIN	TYP	MAX	UNITS	TEST CONDITIONS
V_{IL}	**Input Low Voltage** XML, XMH, TE1, I/O pins XI1 RESET	 -0.1 -0.1 -0.1		 0.75 0.2 V_{DD} 0.60	 V V V	
V_{IH}	**Input High Voltage** XML, XMH, TE1, I/O pins XI1 RESET	 2.5 0.7 V_{DD} 3.0		 Vdd+0.3 Vdd+0.3 Vdd+0.3	 V V V	
V_{OL}	**Output Low Voltage** I/O pins			 0.5	 V	 I_{OL}= 2.0 mA
V_{OH}	**Output High Voltage** I/O pins	 4.0			 V	 I_{OH}= -2.0 mA
I_{IL}	**Input Leakage Current** XML, XMH, TE1, I/O pins XI1 RESET AIN0, AIN1, SH		 <1 <1 <1 <1	 5 5 5 5	 uA uA uA uA	 $V_{ss}<V_{pin}<V_{dd}$ $V_{ss}<V_{pin}<V_{dd}$ $V_{ss}<V_{pin}<V_{dd}$ $V_{ss}<V_{pin}<V_{dd}$
C_i	**Input Pin Capacitance**		6		pF	
I_{cc1}	**Supply Current, Operating**		7	20	mA	Hi-Z outputs
I_{cc2}	**Supply Current, Quiescent**		300	600	uA	Hi-Z outputs
R_{PU}	**Internal Pull-up Resistance, I/O pins**		4.5, 200, Hi-Z		kOhms	Software selected

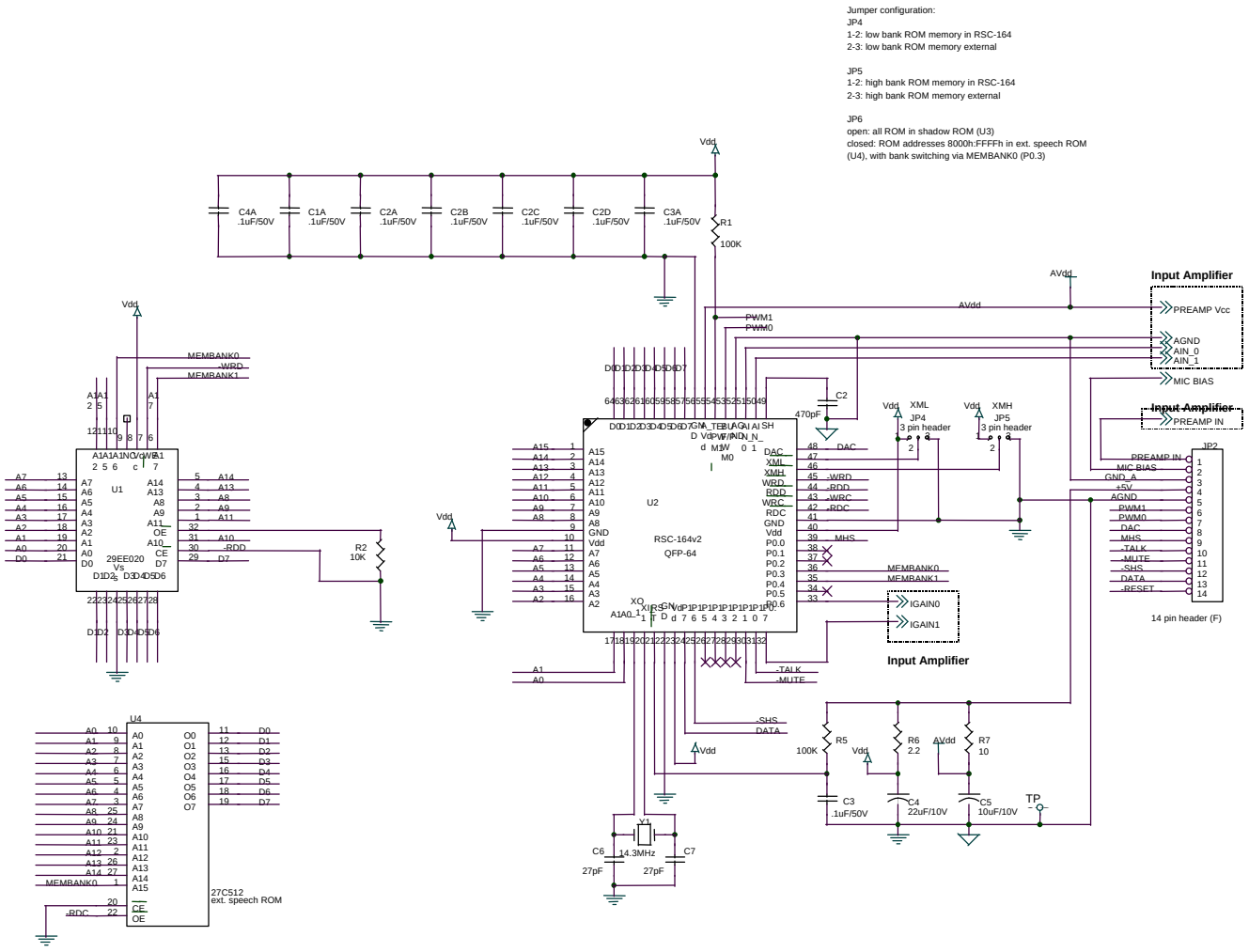
Analog Characteristics

SYMBOL	PARAMETER	MIN	TYP	MAX	UNITS	TEST CONDITIONS
V_{ILA}	AIN0, AIN1	-0.5		0	V	
V_{IHa}	AIN0, AIN1		Avdd/ 2		V	
C_{SH}	SH- capacitance		470 ±10%	pF	V	I_{OL}= 2.0 mA
V_{DO}	DACOUT Voltage Swing	Agnd		Avdd	V	

This page is intentionally blank.

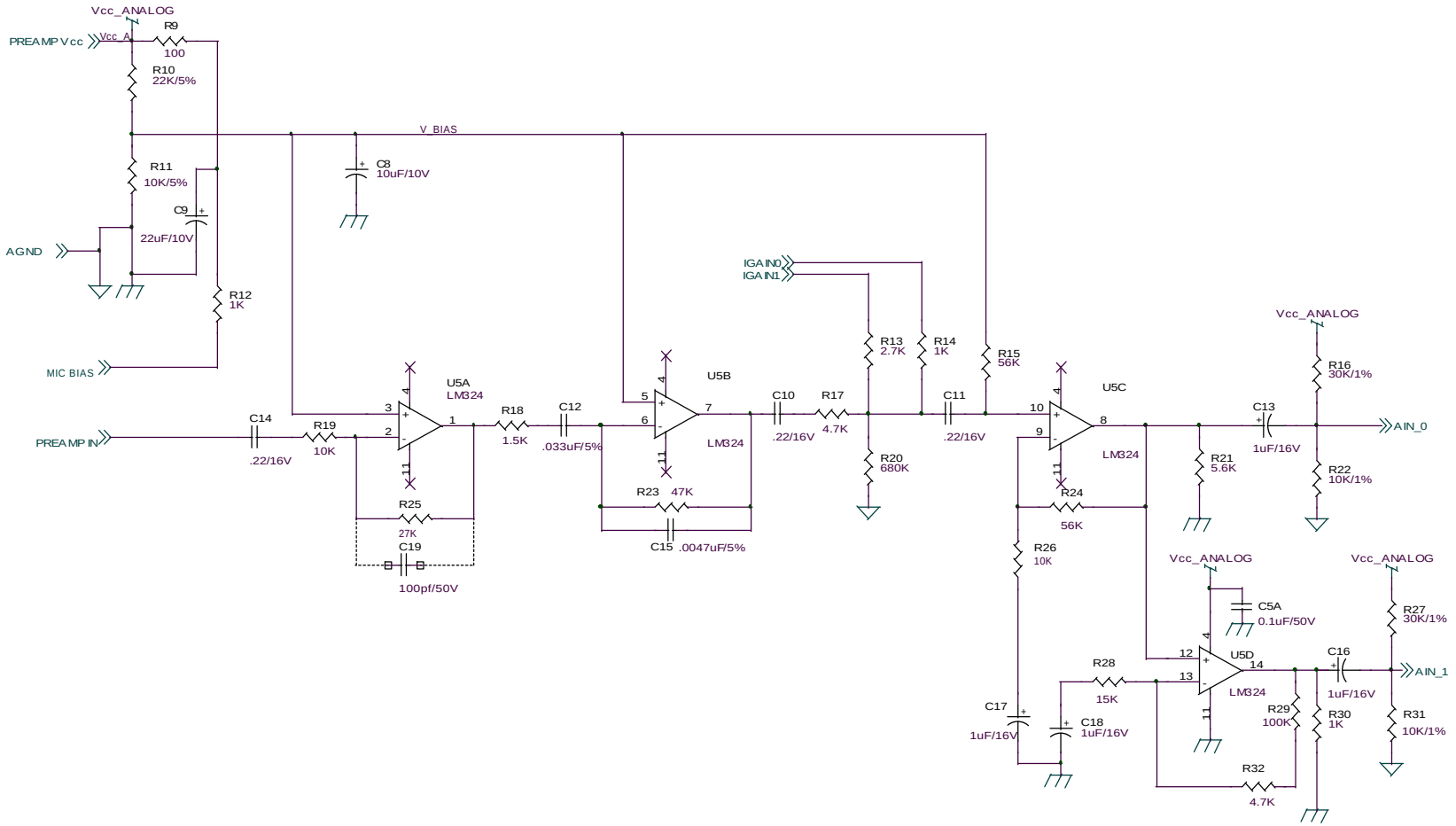
Voice Dialer IC Schematics

Figure 8 - Voice Dialer IC Schematic



This page is intentionally blank.

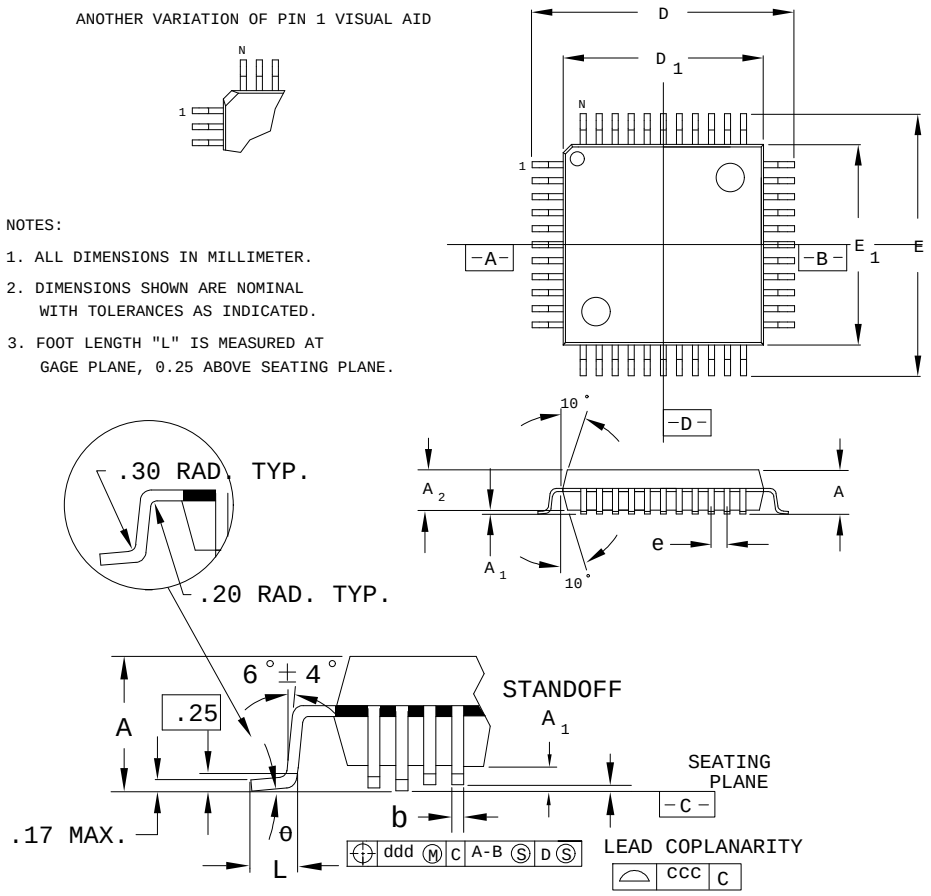
Figure 9 – Input Amplifier Schematic



sensory
voice activation

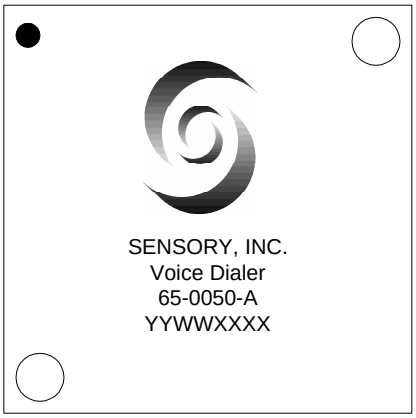
This page is intentionally blank.

IC Metric Drawings



		Body + 3.20 mm FOOTPRINT
Package Thickness		2.00
Dimensions	Tolerance	64L
A	MAX.	2.35
A_1	MAX	0.25
A_2	+.10/-.05	2.00
D	± .25	17.20
D_1	± .10	14.00
E	± .25	17.20
E_1	± .10	14.00
L	+.15/-.10	.88
e	BASIC	.80
b	±.05	.35
θ		0-7°
ddd		.20 NOM
ccc	MAX.	.10

Marking



Pin #1 of: .70 mm
Height of Top Mark: Top Left
Height of Character Logo: 4.475mm
"YYWWXXXX": YYWW - Date Code (year and week)
XXXX - Sensory Identifier

IC Packaging Specification

Package: 64L QFP (14x14x2.0)
Tray: Peak Thin Bakeable, Black, Static Dissipative
84 positions.
Max Bake Temperature: 180° C.
DWG #: ND 1414 2.0 0614 8 Rev A.

Bundle (Typical): 5 + 1 (420 parts)

Packaging Description

Each bundle consists of 5 full trays plus 1 empty tray. Each bundle is strapped with 3 nylon straps. The strappings are approximately at the ¼, ½, and ¾ positions along the long length of the tray. After the trays are bundled, each bundle is placed in a conductive bag with silica gel. The bag is then sealed after air is evacuated from the bag. A label affixed to the outside of the bag contains the following information:

Customer	(S039)*
P.O. No.	
P/Traveller	
Package	(64 LEAD QFP)*
Device/Type	(Voice Dialer)*
Die Lot No.	
Date Code	
Quantity	

*Please Note: Information in ( ) does not change.

The drypacked bags will be individually wrapped in plastic sealed air bubble wrap and placed in a box appropriate to the size of the shipment (8400 = 10 drypacked bags). This box will then be placed into another shipping box and will be packaged with padding (i.e., foam popcorn, sealed air bubble wrap) and labeled accordingly. The shipping box will then be sealed with 2" thick fiberglass reinforced tape.

Dry Bag Recommendations

This device has been qualified to meet JEDEC Moisture Sensitivity Level 3 requirements. Level 3 specifies that the exposure time at the customer site after opening the units is 168 hours in an environment less than 30C and 60% RH.

Manufacturing Information

Manufacturer: Taiwan Semiconductor Manufacturing Corporation (TSMC)
Country of Origin: Taiwan

Voice Dialer Module Specifications

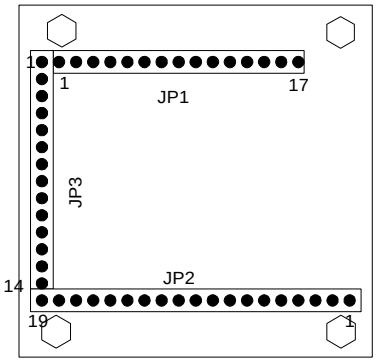
The Voice Dialer Module is a pre-configured circuit that integrates required external components for prototype development. The Voice Dialer Module comes with all of the required external components except for a microphone and speaker. The Voice Dialer Module integrates the following:

- Preamplifier
- Flash Memory
- Atmel AT29C010/SST 29EE010 (1 Mbit) or
- SST 29EE020 (2 Mbit)
- Oscillator (14.32 MHz)
- A location for an optional external ROM (for customized speech or non-English language)

The Voice Dialer Module is 2 inches x 2.875 inches.

What follows are Voice Dialer Module pin descriptions, schematics and a bill of materials.

Module Pin Descriptions



Voice Dialer™ Module

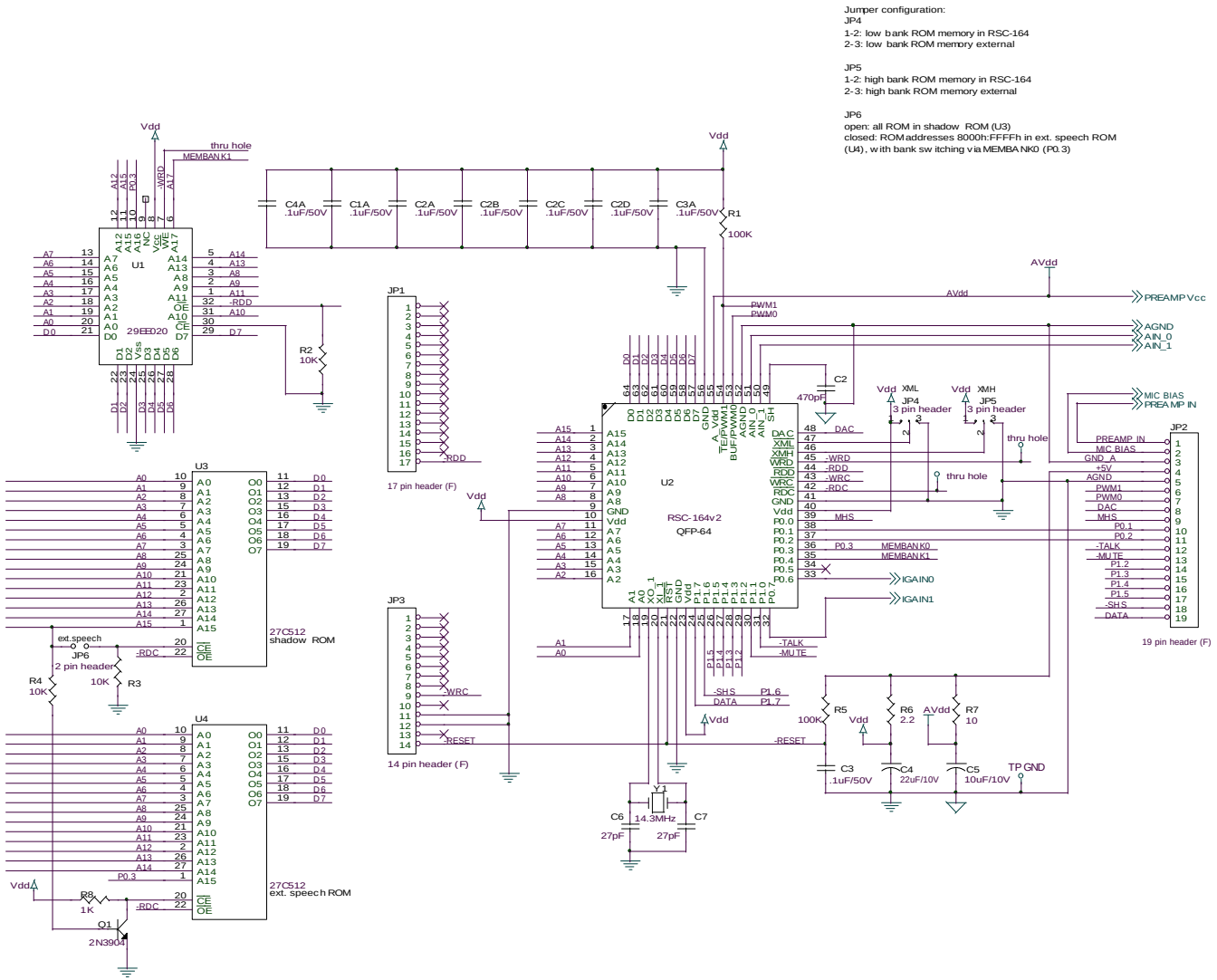
Table 15 - Pin Identification for Voice Dialer Module

Name	Module	Description	I/O
GND	JP3 - 11,12	Digital Ground, CPU core (pins 1 and 33) and I/O (pins 18and 52)	-
AGND	JP2 - 3,5	Analog Ground. For noise reasons, analog and digital grounds should connect together only at the VDS	-
-RESET	JP3 - 14	Reset	I
PWM0	JP2 - 7	Pulse Width Modulator Output0	O
TE1/PWM1	JP2 - 6	Test Mode *or* Pulse Width Modulator Output1 (multiplexed)	I *or* O
DACOUT	JP2 - 8	Analog Output (unbuffered).	O
-MUTE	JP2 - 13	Audio mute signal. Active low during DTMF output	O
-TALK	JP2 - 12	Audio talk signal. Active low during speech synthesis	O
-RDD	JP1 - 17	External Data Read Strobe	O
-WRC	JP3 - 9	External Code Write Strobe	O
MHS	JP2 - 9	Master Handshake. Driven by host.	I
SHS	JP2 - 18	Slave Handshake. Driven by VDS.	O
DATA	JP2 - 19	Serial Data between Master and Slave. Bi-directional	I/O
PREAMP IN	JP2 - 1	Microphone Input Connection	
MIC BIAS	JP2 - 2	Mic Bias (Electret microphone)	
+5V	JP2 - 4	5 Volt (+) Power Supply Connection	
P0.1, P0.2, P1.2, P1.3, P1.4,, P1.5	JP2 -10, 11, 14, 15, 16, 17	Reserved Port Pins, No Connections should be made here	
	JP1 - 1-16, JP3 - 1-8, 10, 13	Unused	-

This page is intentionally blank.

Voice Dialer Module Schematic

Figure 10 – Voice Dialer Module Schematic



This page is intentionally blank.

Bill of Materials for Voice Dialer Module

Table 16 – Bill of Materials for Voice Dialer Module

Quantity	Reference	Part
9	C1A,C2D,C2C,C2B,C2A, C3A, C3,C5A,C4A	.1uF/50V
1	C2	470pF 50v
2	C4,C9	22uF/10v
2	C5,C8	10uF/10V
2	C6,C7	27pF 50v
3	C10,C11,C14	.22/16V
1	C12	.033uF/16v
4	C13,C16,C17,C18	1uF/16V
1	C15	.0047uF/50v
1	C19	100pf 50v
1	JP1- 17P	17P F header
1	JP2 -19P	19P F header
1	JP3 -14P	14P F header
2	JP5,JP4	3 pin header
1	JP6	2 pin header
1	Q1	2N3904
3	R1,R5,R29	100K
6	R2,R3,R4,R11,R19,R26	10K
1	R6	2.2
1	R7	10
4	R8,R12,R14,R30	1K
1	R9	100
1	R10	22K/5%
1	R13	2.7K
2	R15,R24	56K
2	R16,R27	30.1K/1%
2	R17,R32	4.7K
1	R18	1.5K
1	R20	680K
1	R21	5.6K
2	R22,R31	10K/1%
1	R23	47K
1	R25	27K
1	R28	15K
1	U1	29EE020
1	U2	RSC-164v2
1	U3	M27V512-120
1	U5	LM324M
1	Y1	14.3MHz

SENSORY

521 East Weddell Drive
Sunnyvale, CA 94089

TEL: (408) 744-9000
FAX: (408) 744-1299

P/N 80-0121-2