Speech Processing in EMBEDDED APPLICATIONS

Professor Dr Dogan Ibrahim from the Near East University in Nicosia, Cyprus, gives examples of various speech processing chips available in the market, and describes a speech processing example using a microcontroller-based system with a text-to-speech processing chip

SPEECH PROCESSING, especially audio manipulation and sound processing are commonly used in many everyday electronic devices. Some examples are MP3 players, digital voice recorders, speaking GPS receivers, electronic toys, intelligent alarm systems, speaking clocks, radio and televisions, mobile phones, devices used by older and blind people, and many more similar devices.

In general, the speech processing capabilities that can be added to an electronic device are voice recording, voice playback, text-to-speech (TTS) synthesis and speech recognition (SR). Voice recording and voice playback are used in digital voice recorders to store speech in non-volatile memory and then replay it at a later time. Some intelligent recording systems have additional features such as searching for a particular speech, skipping speeches, organising recorded speeches in folders, and so on.

TTS involves reading a written text and converting it into spoken words that can be played through speakers. One typical application of TTS is that the computer can read a piece of text (e.g. via a keyboard or scanner), thus saving the user to look at the screen all the time. People with reading or visual difficulties (e.g. dyslexic or blind people) may find such systems extremely useful.

One of the recent applications of speech processing is in the field of speech recognition (sometimes called voice recognition, or VR), which basically gives a product the ability to "listen and understand". A speech recognition system takes a user's spoken words and interprets what has been said. Speech recognition systems are nowadays commonly used in mobile phones where the numbers to be dialled are read out by the user and the phone rings the required number without the touch of a button. Integration of speech recognition to a product makes that product more intelligent and also more marketable.

Speech Recognition

Speech recognition is mainly divided into two parts: speaker dependent (SD) and speaker independent (SI). Speaker-dependent speech recognition systems are trained by the person who will be using the system and these systems respond accurately only to the person who trained the system. Such systems can achieve 99% accuracy for word recognition and also have large vocabularies.

Speaker-dependent systems also have the advantages that they are language independent and can be trained in any language, or using any type of sound in place of words (for example for people with disabilities who can not speak words). Speaker independent systems are trained to respond to words independent of the user. As a result of this the accuracy of such systems are much lower and also their vocabularies are rather limited compared to speaker dependent systems. Most commercially available speech recognition devices are general purpose speaker independent systems.

The style of speech is also an important factor in most speech recognition systems. Some systems can only recognise words that are isolated from each other and have a short delay between the words. These are the most commonly found systems and low-cost devices, such as toys and simple speech outputting electronic devices, fall into such category. Some more sophisticated speech recognition systems can recognise multiple sets of words when spoken normally and with only natural delay between the words. Continuous speech recognition is the most desirable system as it can recognise natural speech. Such systems are, however, very complex as the words tend to merge with no delay between them and the system has to recognise the words even if they are spoken in a different order. Continuous speech recognition systems are complex but are continually being developed.

Speech Processing ICs

There are many types of commercially available speech processing chips and development kits. Some examples are given here. Magnevation Speakjet is a 20-pin IC designed to add speech and audio to embedded microcontroller applications. The chip is self-contained and requires just an external +5V supply and a speaker for its operation. A mathematical sound algorithm is used to control its five channel internal sound synthesizer to generate vocabulary speech synthesis and complex sound generation. The chip is low cost and is aimed for simple controllers and the hobby market such as robotics, toys and so on.

The Speakjet is programmed with 72 speech elements, 43 sound effects and 12 DTMF touch tones. In addition, sound effects such as the pitch, rate, bend and volume can be controlled. The chip can easily be controlled from a microcontroller.

The TTS256 is a text-to-speech processor chip in a single 28-pin package. This chip is a companion to Magnevation Speakjet and comes with a built in 600 rule database to convert English text to phoneme codes. Speech can easily be generated from ASCII text in microcontroller-based embedded applications, making the chip extremely easy to use in applications where it is required to generate speech. TTS256 is controlled from its serial port and, thus, it is compatible with any microcontroller with a serial port.

TTS-03 is a text-to-speech processor module with serial ASCII input and direct speaker interface. The module operates with a wide supply voltage (+3V to +9V) and has optional IrDA infra-red input for PDA interface. The

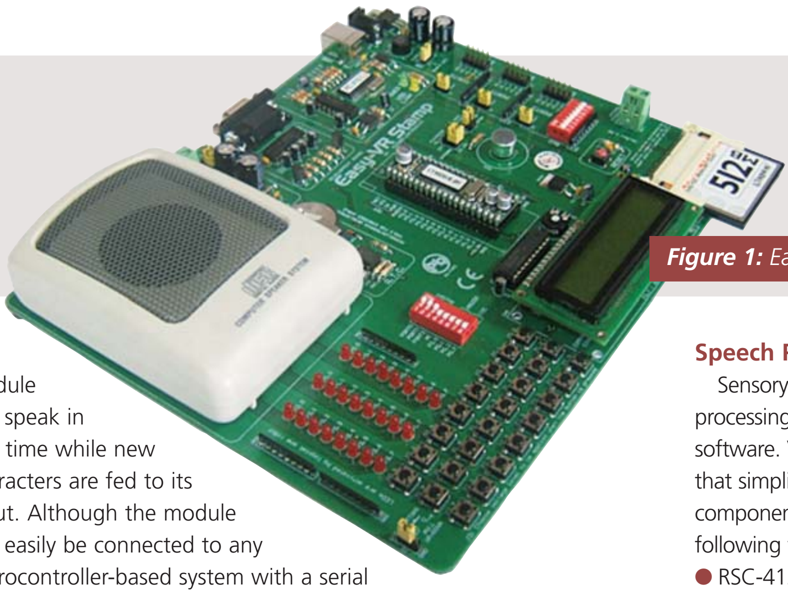


Figure 1: Easy-VR Stamp development kit

module can speak in real time while new characters are fed to its input. Although the module can easily be connected to any microcontroller-based system with a serial output, a mini-motherboard is available with a PS/2 keyboard interface for testing and evaluating the module.

SP03 is a text-to-speech synthesizer chip which also includes an audio amplifier, voltage converter and a PIC microcontroller. Interface to SP03 is via its RS232 serial port, I2C bus interface or parallel interface can be used. The chip can be programmed to generate up to 30 predefined phrases, each phrase having a maximum length of 85 characters. A PC program (SP03.EXE) is available to load these predefined phrases into the chip. To speak any of the predefined phrases a command is sent to the chip in serial or I2C mode, or the required phrase is selected from the 5-bit parallel input port. The volume, pitch and speed of the sound can be selected by commands.

Emic text-to-speech module is based on the Winbond WTS701 speech processor chip and the module can handle values, sentences, numbers and common abbreviations with simple string sentences. The module is manufactured by Grand Idea Studio and distributed by Parallax. The module requires just a +5V supply and an external speaker for its operation. It is controlled from a serial TTL interface and in addition to speech output, the volume, speed and pitch of the sound can be changed by commands, and new abbreviations can be added or an existing abbreviation can be deleted. This module is used in many hobby electronic applications, including in robotics and electronic toys.

S1V30120 is another text-to-speech chip aimed for embedded applications, such as home appliances and office/industrial equipment. The chip is controlled from its serial port and requires an external crystal and an audio amplifier for speaker output. S1V30120 supports several languages and has nine predefined voices.

SSG01 is a sound coprocessor chip in an 18-pin DIP package. It is a 6-voice electronic music synthesizer, sound effects and voice synthesizer chip. This is a low-cost chip aimed for use in manufactured electronics, educational and home projects. SSG01 has a serial port connection and it can easily be connected to an embedded microcontroller with a serial port. The chip requires an external crystal and an audio amplifier to generate sound from a speaker.

HM2007 is a speech recognition chip where up to 40 words can be recognized by the chip and a multiple chip configuration is possible in complex applications. The chip is operated from a single +5V supply and a microphone can be connected directly to the chip.

ISD17120PY is a sound record and playback chip (or a chipcorder) that can be used to record and then play sound for 120 seconds. Different models of the chip can be used for shorter or longer times.

Speech Processing Modules and Boards

Sensory Inc is one of the leading companies manufacturing speech processing chips and hardware modules, and developing speech processing software. VR Stamp is a highly sophisticated module from the company that simplifies the design of speech processing by incorporating all key components into a convenient 40-pin DIP module. The module has the following features:

- RSC-4128 speech processor
- 1Mbit flash and 128kb serial EEPROM
- 14MHz and 32kHz clocks
- 24 I/O lines
- Microphone preamplifier
- Pulse width modulator for speaker output.

VR Stamp module supports the company's FluentChip technology and sound processing library with the following features:

- Speaker-independent and speaker-dependent speech recognition
- 2.4-7.8kbps speech synthesis and sound effects
- Many language modes for international use
- 8-voice MIDI compatible music synthesis
- Audio wakeup from sleep
- Touch tone (DTMF) output.

Some companies offer general purpose digital signal processing chips that can be programmed and used in embedded speech processing applications. For example dsPIC (www.microchip.com) digital signal controller (DSC) chips from Microchip are high performance 16-bit signal processing chips that can easily be used in sound processing as well as in image processing applications. Similarly, Texas Instruments (www.ti.com) TMS series chips, or Analogue Devices (www.analog.com) ADSP series chips can easily be used in speech and image processing applications.

Perhaps one of the easiest ways to learn how speech processing can be done in practice is to use a speech processor development board. The RSC-4x Demo/Evaluation Toolkit from Sensory is a speech development board that helps the user to be familiar with the current speech processing technology and also learn how to program Sensor's speech processing products.

Easy-VR Stamp (see **Figure 1**) is one of the popular speech processor boards manufactured by mikroElektronika. This board is based on Sensory's highly successful VR Stamp module. The board is very low cost and is a good starting point for a student or a professional who wishes to learn the capabilities of the VR Stamp module or to develop speech processing software using the C language and the FluentChip library of sophisticated speech processing functions. The Easy-VR Stamp module has the following specifications:

- Support for VR Stamp speech processing modules (DIP40 socket)
- Real-time clock chip with battery
- On-board speaker and microphone
- Audio amplifier
- 24 LEDs
- 24 buttons
- USB 2.0 programmer

- LCD display
- Compact Flash card slot
- External speaker connection
- RS232 port
- Power from USB port.

The board is normally connected to a PC via its USB bus. Power for the board is derived from this bus and, in addition, the VR Stamp chip is programmed from the PC via the USB bus.

A C-language based compiler, named as RSC-4x mikroC, and chip programmer software named as VR Stamp Loader are also given as part of the distribution. The RSC-4x compiler has no built-in assembler or a linker and, thus, in addition to the supplied software, the user should get a copy of the Sensory MCA-SE assembler and MCLINK linker from the Sensory Inc web site. In addition, for complex speech processing, music and voice recognition applications, it will also be necessary to obtain a copy of the FluentChip library from the Sensory Inc web site.

In this article we will use the Emic chip in a simple text-to-speech application to show how easy it is to include speech in embedded applications. The example given is a "talking thermometer", where the temperature will be read from an analogue sensor IC and will be output as speech every minute.

Digital Audio Effects

Before looking at the design of our "speaking thermometer", it is worthwhile to review the theory of some of the commonly used digital audio effects. Digital audio (sound or speech) processing is basically a branch of the well-known field of digital signal processing (DSP). In a typical digital audio processing application the audio signal is normally received by a microphone, converted into digital format and then fed into a digital processor. The digital processor runs a program where most of the processing takes place. The processed digital audio signal is either stored in the memory of the system as a digital signal, or it is converted into analogue format and is sent to a speaker as an output.

Some simple but commonly used digital audio effects are: audio delay, audio echo, audio equalizer and audio reverberation.

The highly popular MATLAB software (www.mathworks.com) package can be used for experimenting and learning various sound processing functions and algorithms as the package contains a very large number of sound processing functions. This approach is extremely useful, especially for students who may want to learn the basics of digital sound processing. For example, a ".wav" sound file "music.wav" can be read into MATLAB with the function:

```
[mytest, fs] = wavread('music.wav');
```

where fs is the sample rate (in Hertz) used to encode the data in the file. The sound can then be played using the "soundsc" command:

```
soundsc(mytest, fs);
```

where array "mytest" stores the stereo sound and "fs" is the sampling frequency. The two channels of the signal can be separated into two columns using "left" and "right" commands:

```
Left = mytest(:,1);
Right = mytest(:,2);
```

The left channel sound can be played as:

```
soundsc(Left, fs);
```

and the right channel as:

```
soundsc(Right, fs);
```

We can perform a simple processing on the sound files for example by reversing the elements of the sound vector:

```
NewLeft = flipud(Left);
```

and then play the sound in reverse order:

```
soundsc(NewLeft, fs);
```

We can change the sampling frequency and play the sound slower:

```
soundsc(Left, fs/2);
```

or play it faster:

```
soundsc(Left, fs*2);
```

In the following example, the sound file is opened and only the first two seconds sound is played:

```
[mytest, fs] = wavread('music.wav');
nsamples = 2 * fs;
[mytest, fs] = wavread('music.wav', nsamples);
sound(mytest, fs);
```

The following example shows how the sound wave in the file can be plotted (see **Figure 2**):

```
time = (1 / fs)*length(Left);
v = linspace(0, time, length(Left));
plot(v, Left);
xlabel('Time (sec)');
ylabel('Signal strength');
```

Figure 2: *Plot of the sound wave*

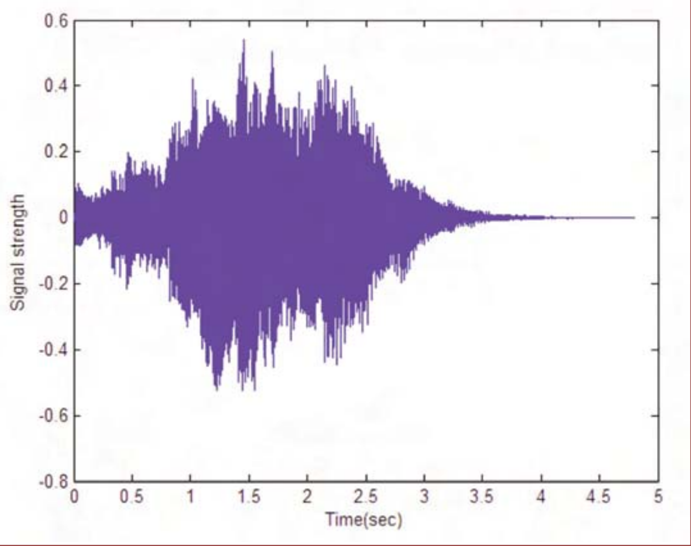
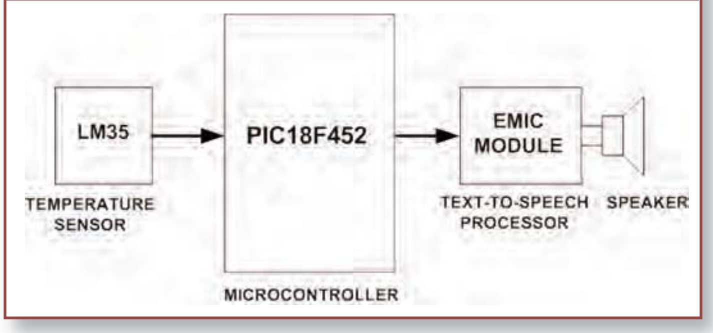


***Figure 3:** Block diagram of the "speaking thermometer"*



Finally, the following example shows how echo can be added to the sound file by adding echo to each sample. In this example V is the new vector that stores the new sound where samples from the original sound are added to the new sound after 1/2 second delay:

```
[mytest, fs] = wavread('music.wav', nsamples);
Left = mytest(:,1);
v = Left;
num = nsamples / 2;
for j = num + 1:length(Left)
        v(j) = Left(j) + Left(j – num);
end
soundsc(v, fs);
```

As can be seen from the above examples, students can easily experiment and learn the basic principles of sound processing by actually modifying the sound characteristics and hearing the resulting effects on the PC speaker.

Example Microcontroller Speech Application

An example is given to show how speech output can be added to a simple microcontroller-based application. In this application an electronic "speaking thermometer" is designed which tells the temperature every minute in the format:

The temperature is nn degrees centigrade

Figure 3 shows the block diagram of the thermometer. A PIC18 microcontroller reads the temperature from an analogue temperature sensor every minute and sends commands to an Emic text-to-speech processor module to tell the temperature via a speaker.

The circuit diagram of the thermometer is shown in **Figure 4**. An LM35DZ type analogue temperature sensor is used in the design and is connected to the analogue port RA0 of a PIC18F452 type microcontroller. LM35DZ can measure the temperature between 0-70°C and its output voltage is directly proportional to temperature, i.e. Vo = 10mV/°C. Thus, for example, at 10°C the output voltage is 100mV and at 30°C the output voltage is 300mV.

The Emic Module

Before describing the microcontroller program it is worthwhile to look at details of the Emic text-to-speech processor module.

Emic (see **Figure 5**) is a 16-pin module (not all pins are used) to provide speech from a written text string. The module is operated from a +5V supply and can drive an 8 ohm, 300mW speaker directly. Text to be spoken can be sent as commands, either in ASCII format or as hexadecimal command sequences. Interface to the module is via a pair of TTL compatible serial wires. The default communication speed is 2400 baud, 8 bits and no parity.

Emic module has the following pins:

VCC	-	+5V supply
GND	-	Ground
BUSY	-	Emic is busy, either receiving characters, or converting text to speech
SIN	-	Serial TTL compatible input
SOUT	-	Serial TTL compatible output
RESET	-	Emic reset pin (active low)
AOUT	-	analogue speech output for external amplification
SP-	-	Negative speaker connection
SP+	-	Positive speaker connection
AIN	-	Analogue input (for the AOUT or SP-/SP+ outputs)

***Figure 4:** Circuit diagram of the thermometer*

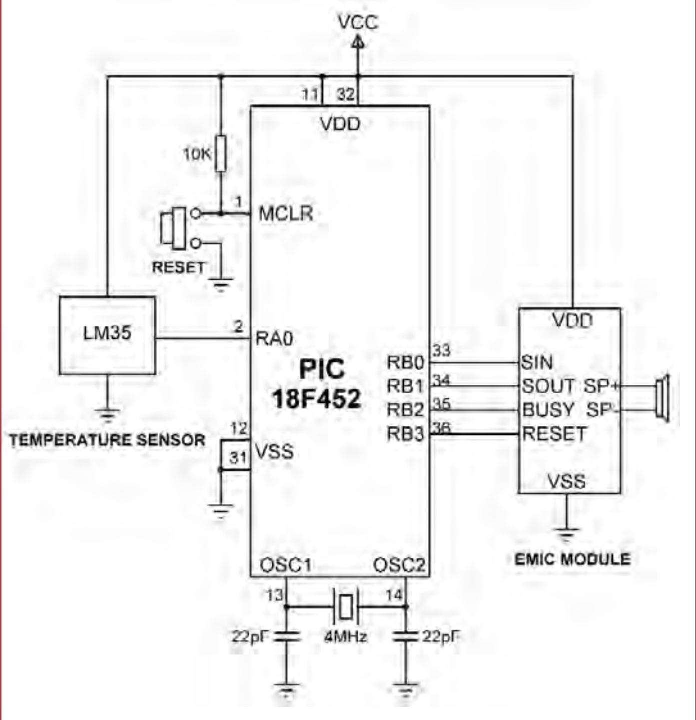
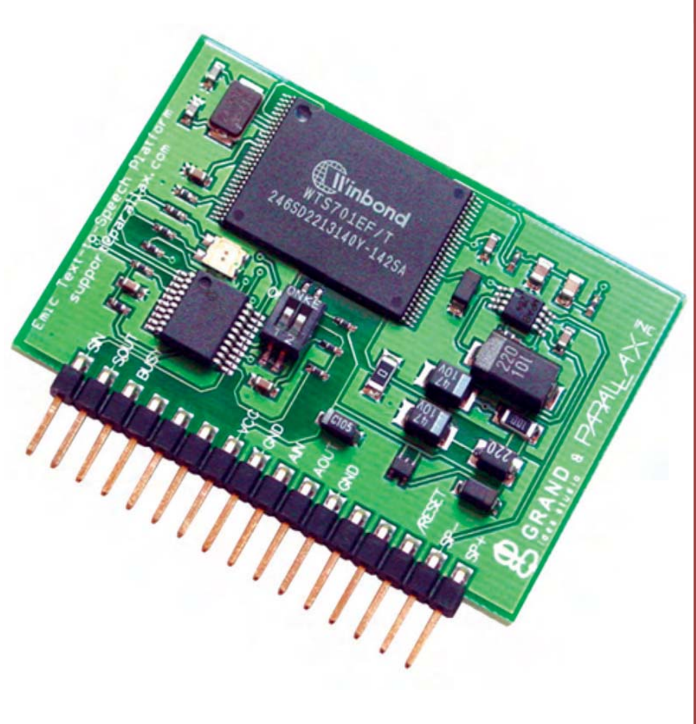


***Figure 5:** Emic text-to-speech processor module*



The microcontroller is connected to the Emic module via the following pins:

Microcontroller	Emic	Description
RB0	SIN	serial input
RB1	SOUT	serial output
RB2	BUSY	Emic busy signal
RB3	RESET	reset to Epic

Two DIP switches are located on the Emic module: SW1 determines the text mode and if set to ON then the module will accept commands in ASCII format. If SW1 is set to OFF, the module will accept commands in hexadecimal format. In this example SW1 is set to ON position.

Switch SW2 controls the host echo mode and if set to ON then each byte sent to the module will be echoed back to the host processor. If SW2 is set to OFF, the module module will not echo each byte to the host (it will only echo "OK" or "ERROR"). In this example SW2 is set to OFF position.

The Emic module has a 128 circular buffer and any command and text data sent to the module must be less than 128 bytes. A text must not be greater than a maximum length of 53 characters, otherwise it will be truncated. End of the user buffer is assumed when the termination character ";" (semicolon) is received in ASCII mode, or 0xAA in hexadecimal mode. In ASCII mode, if a valid command and data are received and the operation is successful, the Emic module will respond with "OK" (or 0x55 in hexadecimal mode). If an incorrect command or data is sent to the module or if the operation has failed, the Emic module will respond with "ERROR" (or 0xFF in hexadecimal mode).The example in this article uses the ASCII mode as it is easier to understand. **Table 1** gives a list of the Emic module command set when operating in ASCII mode. Some of the useful commands are summarised here:

say=text;	-	convert text to speech
volume=n;	-	set the volume. The volume can be 0 to 7 where n = 0 is softest and corresponds to -28dB, and n = 7 is loudest and corresponds to 0dB. Default setting is 4. The range can be incremented or decremented by 1 step using "+" or "-".
speed=n;		set the speed. The speed can be 0 to 4 where n = 0 is the slowest and n = 4 is the fastest. Default setting is 2. The range can be incremented or decremented by 1 using "+" or "-".
pitch=n;		set the pitch. The pitch can be 0 to 6 where n = 0 is the lowest and n = 6 is the highest. Default setting is 1. The range can be incremented or decremented by 1 using "+" or "-".

There are other commands to add abbreviations, delete abbreviations, list existing abbreviations, get the Emic module version number, soft reset the device, analogue audio input and a help command.

***Table 1:** Emic module command set (in ASCII mode)*

Command	ASCII sent
Convert text to speech	say=text;
Set speech volume	volume=n;
Set speech speed	speed=n;
Set speech pitch	pitch=n;
Add abbreviation	addabbr=abbr,text;
Delete abbreviation	delabbr=abbr;
List abbreviations	listabbr;
Version numbers	version;
Soft reset	reset;
Audio input	audio;
Help	help; (or ?;)

Program Description

Figure 6 shows the operation of the thermometer project in PDL. The actual program listing is given in **Figure 7**. In this example the program was developed using the mikroC compiler (www.mikroe.com). At the beginning of the program port I/O directions and the analogue port are configured and the Emic module is reset. Then the serial port is configured to operate at 2400 baud and the welcome message "welcome to speaking thermometer" is sent to the speaker via the Emic module. The main loop of the program in enclosed inside a 'for' loop which executes forever with one minute delay (by calling to procedure Wait_One_Minute) between each loop, and reads the temperature by calling to procedure Read_ Temperature. The temperature is converted into ASCII using built-in function ByteToString and is stored starting at offset 19 of array TempMsg. Text string TempMsg is then sent to the speaker by calling to procedure Speak.

***Figure 6:** Operation of the speaking thermometer*

```
BEGIN
        Configure I/O ports
        DO FOREVER
                Read temperature from A/D converter
                Convert to Centigrade
                Store temperature in a text string
                Send text string to EMIC module
                Wait 1 minute
        ENDDO
END
```

```
/*-------------------------------------------------------------------------------------
-------------
 SPEAKING THERMOMETER
         ====================
This is the software for the speaking thermometer project.
A PIC18F452 type microcontroller receives the temperature
from a LM35DZ type analogue sensor connected to port pin
RA0. The temperature is sent to a speaker via an EMIC text-
to-speech processing chip.

The temperature is output to the speaker in the following
format:

The temperature is nn degress centigrade

The microcontroller is operated with a 8MHz crystal The
connection between the microcontroller and external world
is as follows:
 RA0-> Vout (LM35DZ)
 RB0-> SIN (EMIC)
 RB1-> SOUT (EMIC)
 RB2-> BUSY (EMIC)
 RB3-> RESET (EMIC)

File:   THERMOMETER.C
Date: September, 2009
---------------------------------------------------------------------------------------
-----------*/
#define TX_Pin 0
#define RX_Pin 1
#define RESET PORTB.F3
#define BUSY PORTB.F2

char WelcomeMsg[] = "welcome to speaking thermometer";
char TempMsg[] = "the temperature is nn degrees
centigrade";

//
// Send a text to EMIC text-to-speech processor module
//
void Speak(char *ptr)
{
 char text[53],i, j;
 text[0]='s'; text[1]='a'; text[2]='y'; text[3]='=';

 j = 4;
 while(*ptr)
 {
 text[j] = *ptr++;          // Insert message
 j++;
 }
 text[j]=';';               // Insert terminator, ";"

 for(i=0; i<=j; i++)        // Send to EMIC module
 {
 while(BUSY);  // Wait if Emic is busy
 soft_Uart_Write(text[i]);          // send to serial port
 }
}

void Wait_One_Minute()
{
 unsigned char i,j;
 for(j = 0; j < 60; j++)
 {
 for(i = 0; i < 4; i++)Delay_Ms(250);
 }
}

void Read_Temperature(void)
{
 unsigned long Vin;
 unsigned char Temperature,j;
 char txt[4];
 Vin = Adc_Read(0);                  // Read temperature
 Vin = Vin * 5000 / 1024;                    // Convert to mV
 Vin = Vin / 10;                     // Convert to Celsius
 Temperature = Vin;                  // As byte
 ByteToStr(Temperature, txt);        // Convert to string
 for(j=0; j<3; j++)TempMsg[19+j] = txt[j];     // Add to
TempMsg
}

void main(void)
{
 TRISA = 1;              // RA0 is input
 ADCON1 = 0x80;                      // Configure A/D
 TRISB = 6;              // RB1,RB3 are inputs
 RESET=0;                // Reset EMIC module
 Delay_ms(200);
 RESET = 1;

//
// Configure serial port to 2400 Baud
//
 Soft_Uart_Init(PORTB, RX_Pin, TX_Pin, 2400,0);
//
// Send welcome message
//
 Speak(WelcomeMsg);
//
// Program loop. Read temperature and send to speaker
continuously
//
 for(;;)
 {
 Read_Temperature();                 // Read temperature
 Speak(TempMsg);                     // Send to speaker
 Wait_One_Minute();                  // Wait for 1 minute
 }
```

Figure 7: *Program listing of the speaking thermometer*

An Important Topic

Speech processing, such as speech recognition and text-to-speech, is becoming a very important topic in intelligent embedded applications. Nowadays most mobile phones include some form of speech recognition modules to help use the phone hands-free. Another very important application of speak recognition is to help people with disabilities. For example, a lift with a speech recognition processor can help a blind person to control the lift by speech. Similarly, an elderly person for example can turn a kettle ON or OFF by speech.

Text-to-speech is largely used in alarm and control systems with audio output and in hobby electronics, such as robotics and electronic toys. A text-to-speech processor based system can also help a person with reading and learning difficulties, for example by reading a book or a newspaper, or any other document.

This article has given examples of various commercially available speech processing chips and modules. An example text-to-speech application is given in the form of a "speaking thermometer" to show the simplicity and the beauty of including speech in microcontroller-based systems. ■