SWEET-TALKING BEEB

Vincent Fojut lends an ear to a low-cost speech synthesiser

'Sweet Talker' speech synthesiser, Cheetah Marketing, 24 Ray Street, London EC1R 3DJ, tel: 01-833 4909, £24.95

CHEETAH'S *Sweet Talker* speech synthesiser at £24.95 must surely be the most inexpensive way of adding synthetic speech to the Beeb. So just what do you get for the price, and how does the product shape up?

The hardware could hardly be simpler and less obtrusive. A small circuit board of some 3in by 2in holds the speech synthesiser chip (the popular General Instruments SP0256), and a modicum of external components.

The whole assembly plugs into IC99 on the Beeb's main circuit board via an IC socket mounted on extension pillars. The unit requires a fair amount of pressure to be driven home, and I would recommend that users unscrew the main circuit board from the Beeb's casing, so that it may be supported from the underside during insertion. This helps minimise the risk of hairline cracks appearing in the PCB tracks.

Since IC99 is very close to the ribbon cable connecting the keyboard to the BBC's circuit board, space is a little tight, but it should be possible to install the unit without exerting undue force on any component.

There are no tracks to cut, soldering to be done or awkward flying leads to connect. In fact, being so small, the speech synthesiser should not impinge on other hardware fitted inside the Beeb, such as RAM/ROM expansion boards, but check first!

The software accompanying the package consists of a cassette-based program, 'Beebtalk'. Not only does this provide an entertaining demonstration of the *Sweet Talker's* capabilities, but the code is also meant to serve as an example of how to program the speech chip.

To generate speech, *Sweet Talker* uses an approach based on allophones, which can be considered as the fundamental building-blocks of speech. When fed with a number between 0 to 63, the synthesiser chip can generate one of 59 distinct allophones, plus 5 different periods of silence.

The allophone technique used in Cheetah's synthesiser does not produce particularly natural-sounding speech, as human speech is not, unfortunately, a series of discrete sound segments. The advantages, however, are that it is easy to program, very cheap, and should allow any English-language phrase to be synthesised.

In fact, the demonstration software goes even further, with humorous results. In addition to 'speaking' in English, the program attempts three other languages — French ('Bonjour'), German ('Guten Tag'), and that other well-known 'foreign' tongue, Scottish (yes, the inevitable 'See you, Jimmy'l).

The 'Beebtalk' program includes a short machine-code routine, together with examples of how to call it from Basic, to generate the words of your choice. It works in the following way. Take your phrase and break it into its constituent allophones, then convert these into their corresponding numeric codes, and embed the string of numbers in a DATA statement. By calling the machine code routine with each number in turn (for example, using a FOR . . . NEXT loop), the synthesiser chip should then generate the required sounds.

For example, the phrase 'Acorn User' can be built from the following allophones (using Cheetah's notation):

EY, KK3, OR, NN1, YY2, UW1, ZZ, ER1.

These, in turn, give the following numbers:

20, 8, 58, 11, 25, 22, 43, 51.

In practice, the string would probably be terminated with a 'silence' code, such as 0, which effectively turns the device off. Without this extra code, the synthesiser would continue generating the last sound issued, which can be very disconcerting!

While the above approach does work, it obviously has drawbacks. There is no clear correlation between the numbers used, and the sounds which they produce. Consequently, building up even quite short phrases becomes a tedious exercise of conversion and cross-referencing.

In order to simplify the task of programming the device during evaluation, I wrote a short program to allow input to be made using the allophones themselves, leaving the computer to convert them to the appropriate numbers before feeding them td the speech chip. This was a relatively straightforward exercise, and code from Cheetah's own demonstration program was incorporated (as it is intended to be) with minimal effort. I could then type in a string such as 'HH1EHLLOW' to produce the word 'hello'. Not ideal, but much better than a string of meaningless numbers. I leave it to others to take this one stage further, and write a program to generate speech from 'normal' English text (do not underestimate the enormous complexity of this task!).

As far as speech quality is concerned, the 'voice' is unmistakably synthetic, with a falsely high degree of hissy, white noise in a number of sounds. Certain allophones are particularly dubious — the 'W' for example, just doesn't seem to make the mark. Indeed, the untrained listener may be hard-pressed to distinguish many sounds that the synthesiser produces. The monotonal quality of the voice not only sounds artificial, but also becomes irritating after a while.

Cheetah's demonstration program craftily displays each phrase as the speech synthesiser generates the sounds. In a sense, this is cheating (excuse the pun), since it is much easier to 'understand' the device when you already can see what it is trying to say! A more accurate appraisal of its fidelity can be had by listening to the program (the first few times, especially) without looking at the screen. Most friends I tried this on were unable to make out some of the words at first.

However, intelligibility does improve, not only as one becomes more used to the characteristics of its own particular ' voice', but also as one learns to program the unit more skilfully.

The *Sweet Talker* seems ideal for those wanting a low-risk introduction to speech synthesis, especially where fidelity is not of critical importance. Above all, the device is fun to use, relatively easy to program and provides considerable enjoyment at minimum cost.