ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Implementation of Text-To-Speech for Real Time Embedded System Using Raspberry Processor

Endarapu Vanitha
M.Tech(Embedded Systems)
Sindhura College of Engineering & Technology, Godavarikhani
Karimnagar(Dt), India.

Pradeep Kumar Kasarla
Associate Professor
Sindhura College of Engineering & Technology, Godavarikhani
Karimnagar(Dt), India.

E Kumaraswamy
Lecturer
Sindhura College of Engineering & Technology, Godavarikhani
Karimnagar(Dt), India.

ABSTRACT:
The real time hardware implementation of Text-To-Speech system has been drawing attention of the research community due to its various real time applications. These include reading aids for the blind, talking aid for the vocally handicapped and training aids and other commercial applications. All these applications demand the real time embedded platform to meet the real time specifications such as speed, power, space requirements etc. In this context the embedded processor ARM has been chosen as hardware platform to implement Text-To-Speech conversion. This conversion needs algorithms to perform various operations like parts of speech tagging, phrase marking, word to phoneme conversion and clustergen synthesis. We are using Raspberry Pi, a credit card–sized single-board Processor developed in the UK by the Raspberry Pi Foundation for the implementation.

Index-terms: ***Raspberry Pi Processor, Text-to-Speech(TTS), Reading aids, Talking aid, ARM, Embedded systems.***

I. INTRODUCTION:

An embedded system is a dedicated computer system designed for one or two specific functions. This system is embedded as a part of a complete device system that includes hardware, such as electrical and mechanical components. The embedded system is unlike the general-purpose computer, which is engineered to manage a wide range of processing tasks. Because an embedded system is engineered to perform certain tasks only, design engineers may optimize size, cost, power consumption, reliability and performance. Embedded systems are typically produced on broad scales and share functionalities across a variety of environments and applications.

Text-to-speech (TTS) is the problem of Automatic conversion of text into speech that Resembles, as closely as possible, a native speaker of the language reading that text [1]. In TTS Systems an arbitrary text is converted into speech which can be heard out loud.

TTS conversion involves several steps. Broadly they can be classified as front end and back end steps. In front End steps all the language related processing is done. Some of the front end related steps are text to kenization, text normalization, word to phone Conversion etc. In back end steps speech related Signal processing is done. Some of the back end Steps are pitch marking, duration modeling, speech Synthesis etc. The implementation details of all the Steps for TTS are covered in this paper. The Implementation presented here is a non OS based Implementation. The results of implementation of these steps on arm and a comparison of these results with x86 implementation is presented.

II. SYSTEM ARCHITECTURE:

2.1 BLOCK DIAGRAM:

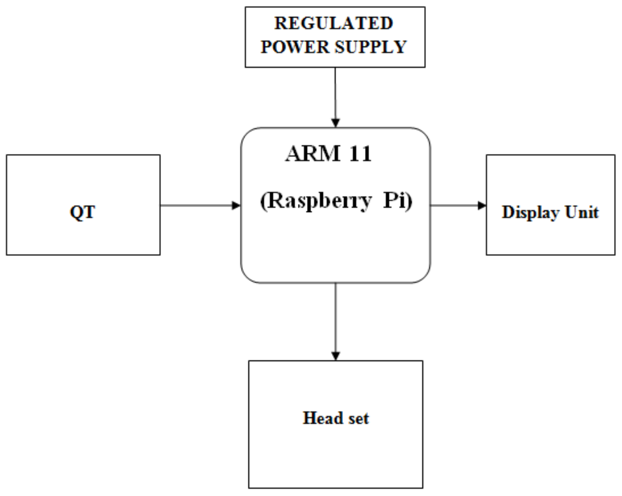


Figure-1: block diagram of project

2.2. EXISTING METHOD

In the existing method if we want to give speech means it is possible only predefined voices can possible and it can store limited voices. This is the drawback of present in existing system. Therefore the users can't get fully information by using this method.

2.3 PROPOSED METHOD

The proposed method is used to overcome the drawback present in existing method. The design of this project involves text to speech. Here whatever we are giving input from any keyboard corresponding output will get in the form of voice means speech. The development board with ARM architecture is selected as the hardware platform. Start-up codes, OS kernel and user's application programs are together stored in a NAND FLASH. Application programs run in 64MB SDRAM, which can also be used as the room of various data and the stack

III. HARDWARE IMPLEMENTATION:

3.1 RASPBERRY PI PROCESSOR:

Your Raspberry Pi board is a miniature marvel, packing considerable computing power into a footprint no larger than accredit card. It's capable of some amazing things, but there are a few things you're going to need to know before you plunge head-first into the bramble patch. The processor at the heart of the Raspberry Pi system is a Broadcom BCM2836 system-on-chip (SoC) multimedia processor. This means that the vast majority of the system's components, including its central and graphics processing units along with the audio and communications hardware, are built onto that single component hidden beneath the 256 MB memory chip at the centre of the board. It's not just this SoC design that makes the BCM2836 different to the processor found in your desktop or laptop, however. It also uses a different instruction set architecture (ISA), known as ARM. A better-quality picture can be obtained using the HDMI (High Definition Multimedia Interface) connector, the only port found on the bottom of the Pi. Unlike the analogue composite connection, the HDMI port provides a high-speed digital connection for pixel-perfect pictures on both computer monitors and high-definition TV sets. Using the HDMI port, a Pi can display images at the Full HD 1920x1080 resolution of most modern HDTV sets.

Figure-2: Raspberry Pi processor

3.2 DESCRIPTION OF PROJECT:

Since decades, real time hardware implementation of Text-To-Speech system has been drawing attention of the research community due to its various real time applications. These include reading aids for the blind, talking aid for the vocally handicapped and training aids and other commercial applications. All these applications demand the real time embedded platform to meet the real time specifications such as speed, power, space requirements etc. In this context the embedded processor ARM (Advanced RISC

Machine), has been chosen as hardware platform to implement Text-To-Speech conversion. This conversion needs algorithms to perform various operations like parts of speech tagging, phrase marking, word to phoneme conversion and clustergen synthesis. These algorithms are coded and developed in C using eclipse IDE and finally implemented on commercially available ARM11 microcontroller. Experiments have been performed on ARM microcontroller using test cases. It has been observed that the performance of the ARM based implementation is very close to x86 implementation. In this way we can design text to speech by using **Raspberry Pi** board and **Embedded Linux**.

ARM11 microcontroller is used for the implementation of the text to speech system. ARM is the most widely used platform for the embedded systems. ARM has been chosen as a platform for the project because of its low power and high speed operations. ARM11 has been specifically chosen for the implementation of text to speech conversion because it has SDRAM memory and it supports image sensor interfacing which could be used for embedded system development for visually challenged readers which is our future scope for research.

IV. EXPERIMENTAL RESULTS:

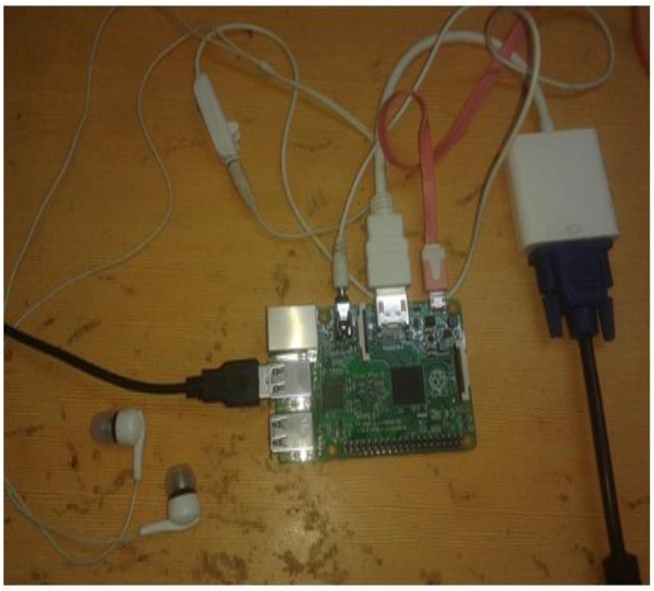
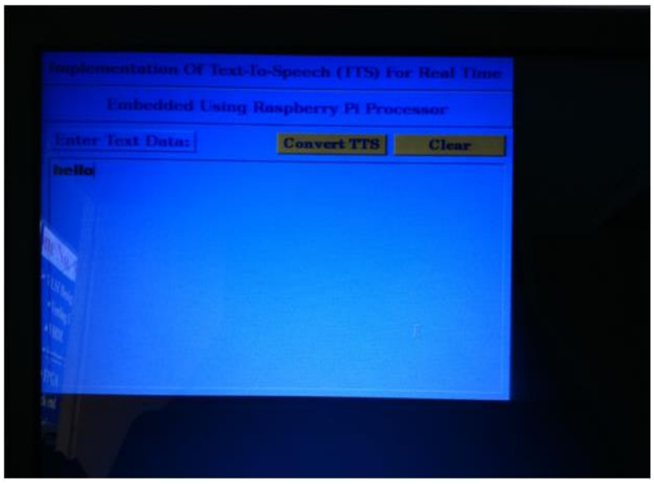


Figure-3: Hardware implementation



Implementation Of Text-To-Speech (TTS) For Real Time

Embedded Using Raspberry Pi Processor

hello

Figure-4: Output of Project

V. CONCLUSION

The paper described a way of implementing the text to speech system on ARM microcontroller. Among different speech synthesizing algorithms CLUSTERGEN synthesis is chosen for implementing a real time text to speech system. The results of the implementation are presented. These results shows that the x86 based implementation and ARM based implementation are very close. From the results it can be concluded that a complete low cost real time embedded system can be built with the implementation presented in the paper. This embedded system can be used in many applications like reading aid for the visually disabled persons, talking aid for vocally handicapped.

VI. REFERENCES

[1] Shin, C. and Sproat, R. "*Issues in text-to-speech conversion for Mandarin*", Computational Linguistics and Chinese Language Processing, pp. 35-82, August 1996.

[2] The FLITE website. [Online]. Available :http://www.speech.cs.cmu.edu/flite/

[3] Taylor, P. and Black, A. "*Assigning Phrase Breaks from part-ofspeech Sequences"* Computer Speech and Language 12, 99-117,1998

[4] DeRose, S. "*Grammatical category disambiguation by statistical optimization*". Computational Linguistics, pp. 31-39, 1988
[5] Black, A., Lenzo, K. and Pagel, V. "*Issues in Building General Letter to Sound Rules"* 3rd ESCA Workshop on Speech Synthesis, pp. 77-80, 1998
[6] Pagel, V., Lenzo, K. and Black, A. "*Letter to sound rules for accented lexicon compression"* ICSLP98, pp 2015-2020, 1998.
[7] Black, A. "*Predicting the intonation of discourse segments from examples in dialogue speech*", ATR Workshop on Computational modeling of prosody for spontaneous speech processing, 1995.
[8] Black, A. "*CLUSTERGEN: A Statistical Parametric Synthesizer using Trajectory Modeling",* Interspeech ICSLP, pp. 1763-1765, 2006
[9] Keiichi, K., Takao, K. and Satoshi, I. *"Speech Parameter Generation From HMM Using Dynamics Features" ,* European Conference on Speech Communication and Technology, 1995
[10] Satoshi, I. "*Cepstral Analysis Synthesis on the MEL Frequency Scale*" , IEEE International Conference on ICASSP, pp. 93-96, 1985