EG40GA

Computer & Software Engineering

Reduced Instruction Set Computer Design

Allan Bruce

CONTENTS

AIM

This purpose of this project is to increase our knowledge and understanding of Computer Engineering. It involves the simple design of a RISC (Reduced Instruction Set Computer) Processor. The design is not explained in depth, but is applied to a specific function; to be used in a portable mp3 (MPEG audio Layer III) player. As such, the processor must be a low power chip and must include necessary instructions for the decoding of mp3 files (decoding will be carried out in software for future compatibility). The chip must also be inexpensive to produce; so the silicon size must remain relatively small. In order to achieve this the design should not be overly complex, since this would increase the physical size of the chip.

INTRODUCTION

In order to design the RISC chip, some knowledge of the mp3 format is needed. Mp3s are digitally stored audio files, similar to those found on compact discs that are compressed, or shrunk in size, to maximise storage capacity. A typical 5-minute Compact Disc Audio track is around 45 Megabytes (MB) in size whereas a CD-quality mp3 file coded with a bit rate of 128kbps (see Appendix) would be approximately 5MB. This size difference is achieved using the advanced algorithms and data storage methods, stated below:

- Minimal Audition Threshold
- The Masking Effect
- Joint Stereo
- Huffman Encoding

These methods of compression are discussed in the Appendix

REVISED REQUIREMENTS SPECIFICATION

The following outlines the requirements that were set and adhered to in designing the processor. The assignment is to design a RISC microprocessor is to be embedded into a range of portable mp3 players. This imposes a power consumption and size restriction so the design needs to be simple, low power and cost effective. Decoding of mp3 files is to be carried out in software using the processors instruction set and therefore must contain necessary instructions for this, either directly within hardware or in the processors microprogram. There will be an in-built ROM to store the necessary software and mp3 files will be transferable via a USB interface. The mp3 player will be operated using a touch sensitive backlit colour LCD screen. This LCD unit will contain its own coprocessor for basic operations and control of the interrupt lines.

Storage of the mp3 files will be accomplished using inbuilt 64Mb of RAM, or by using additional memory in the forms of flash memory, via the compact flash slot. There will need to be enough expansion for future competitiveness.

The mp3 player must also contain a basic operating system that will carry out file transfers and streaming of data.

FUNCTIONAL SPECIFICATION

CPU SPECIFICATIONS

The CPU is based on a 0.18-micron process running at 24.576 MHz. It is a 32-bit RISC processor with high-speed 32-bit data bus. The operating voltage is 2.2V dc

CPU CORE INSTRUCTIONS

All instructions in this design are 32-bit instructions. There are 15 main instructions implemented in this design, which are outlined below. Other instructions will be provided by the system microprogram.

- Data Movement Instructions

  Load, Store and Register-Register move instructions will be implemented as a means of data movement within the processor.

- ALU Instructions

  OR and NOT will be the logics instruction implemented as all other logical operations can be carried out using a series of these gates. COMP (Compare) and LS / RS (Shift) instructions will be provided for data manipulation. Arithmetic operations are available in the form of the ADD, SUB, and NEG (Negation) instructions.

- Branch instructions

  These include jump if zero, jump if less than and unconditional jump as standard. A CALL instruction is also implemented; on execution of this instruction the contents of SP and CP. A RET (return from subroutine) instruction will allow the original values of SP and CP to be updated.

Instruction format is fixed length to maximise speed of program execution.

REGISTERS

- 32 GPRs (General Purpose Registers), each of 32 bits
- One register for state, time, and track name
- Program Counter and Stack Pointer
- Status Register - Bit indications for status ZVCN

ADDRESSING MODES

Compatible addressing modes include:

- Immediate
- Register Direct
- Indirect
- Base + Index

INTERRUPTS

The processor has 4 interrupt lines with priority toward IRQ1

- IRQ1 – Non maskable interrupt (used for emergency power down)
- IRQ2 and IRQ3 – General interrupt lines
- IRQ4 – Clock edge interrupt line

OPERATING SYSTEM

The operating system will not be fully compatible with other operating systems. A simple I/O and file handling system has been designed specifically for the purpose of this mp3 player.

DESIGN DOCUMENT

This section provides information on the areas covered in this design, including, the general architecture of the player, the RISC core, the ALU hardware, addressing modes, instruction format, registers, branch instructions, interrupts and input/output.

GENERAL ARCHITECTURE OF PLAYER

The layout of the player is as shown below (fig 1):

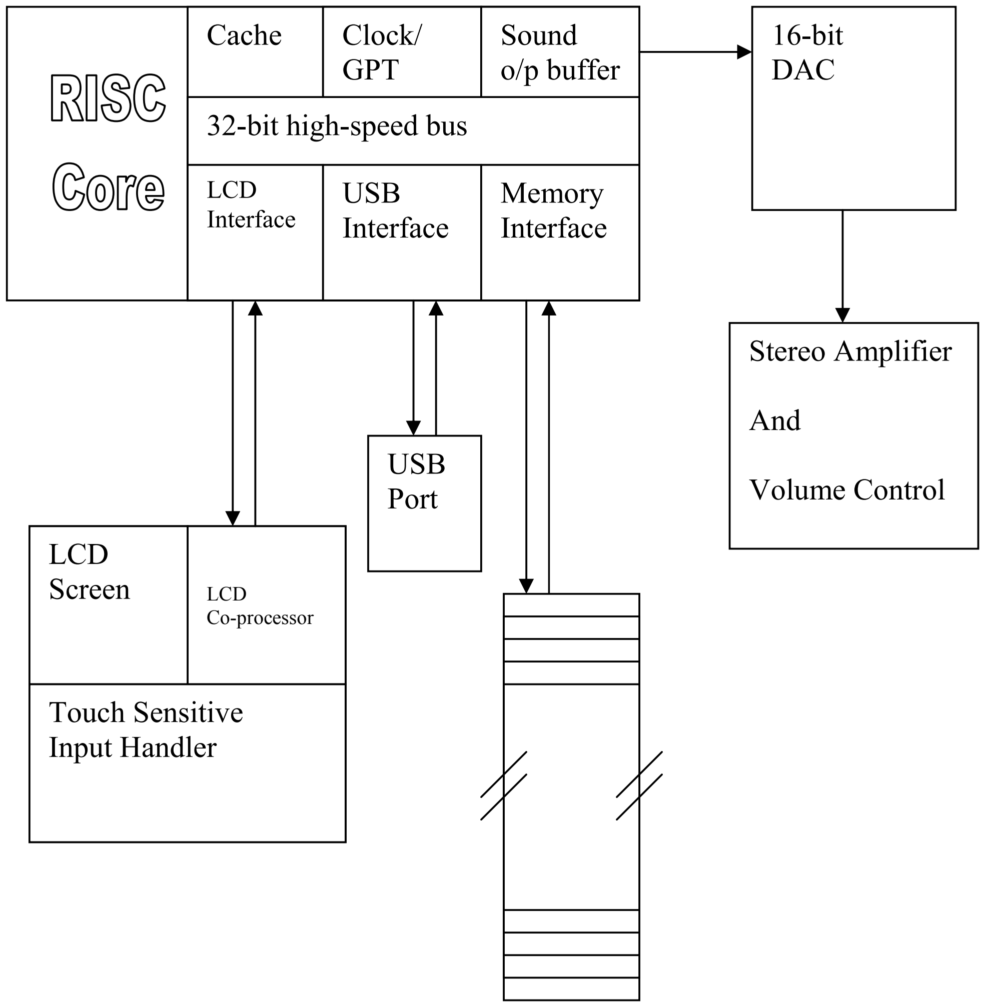

figure 1 - General Architecture of MP3 Player

The high-speed bus allows communications between the RISC core and all external components.

All play functions are selected using the touch sensitive LCD screen. The LCD unit contains its own co-processor to enable interrupts and also display the system state and track name. The LCD interface controls data flow between this and the RISC core.

Once an mp3 file is decoded it is sent to the 16-bit DAC (Digital to Analogue Converter) via the sound output buffer. A stereo amplifier then produces audio at the required volume.

A General Purpose Timer is present to ensure the audio is decoded at the correct frequency. This enables the decoded sound to maintain the quality of the original recording.

The small memory cache allows the most frequent instructions to be accessed as quickly as possible; this significantly improves the performance as mp3 decoding contains a lot of repeated instruction execution.

The memory interface manages data read from and written to the memory. As there is an expansion slot, this unit controls the location of memory that the mp3 files are to be read from.

RISC CORE

The RISC core is as shown below (fig 2)

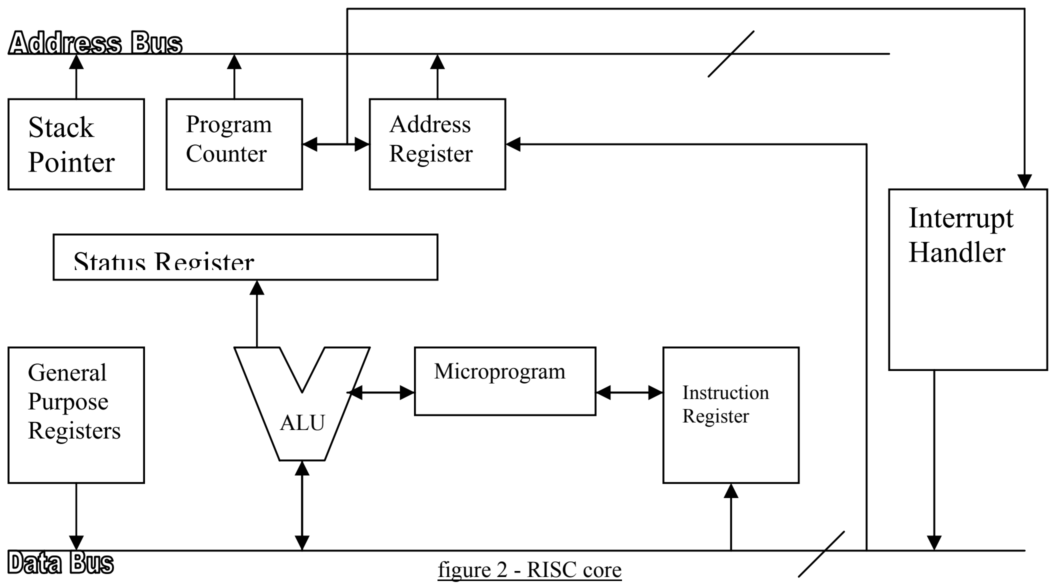

figure 2 - RISC core

This is a typical layout for a RISC CPU. CPUs commonly run at 24.576 MHz for mp3 chips. At this basic level, it is not possible to predict what speed the CPU should run at for this specific design.

ALU HARDWARE

The ALU carries out most operations, using the microprogram to execute complex instructions. Supported instructions are ADD, SUB and NEG. The inclusion of the SUB instruction is to save CPU cycles, as without it, the ALU would have to negate then add. The ALU hardware contains a set of adders for addition and subtraction, which also utilise the carry bit in the status register. Shifting operations are also carried out using the ALU, which can be used for multiplication or division by 2. Logic operations are executed using OR and NOT gates, which allows the microprogram to simulate any logic gates. The microprogram intercepts any instruction that are not hardware implemented and translates them to a format which is hardware compatible, making it possible for the ALU to receive all appropriate instructions and carry them out. All gates may be implemented using NAND gates, however this process requires many more cycles than using OR and NOT gates. Since speed is important in this process, OR and NOT were chosen. OR and NOT gates are now very inexpensive to implement. They use only a little more transistors than a NAND gate would (increase in silicon area is negligible compared to whole chip), so the design will be no more expensive, and will use very little more power than if just NAND gates were used.

The CPU was designed to be 32-bit, which enables it to support the USB facility. This also allows the chip to use simple, single cycle address modes.

ADDRESSING MODES

The immediate address mode is necessary for almost all applications. This is when the operand denotes the required data. It allows quick executions of instructions, but has a limited range of data values.

Register Direct mode allows the program to store operands in GPRs. This is beneficial, as they may contain data that is used many times, and operation is quick. This allows 32-bit manipulation but only a very limited range of data can be processed.

Indirect Addressing is used when the data is contained in an unknown memory location at compile time. This is when the operand points to memory locations that contain the address of the data. This method is relatively slow but has a large data range. Indirect addressing is not necessary but makes the chip easier to produce software for.

The last method implemented in the processor is Base+Index. This is when the Operand points to GPRs and the data contained in the GPRs are the address of the data. There can also be a base added to this for ease of incrementing addresses. This method allows a very large range of data to be used but it is slow in operation. This latter addressing mode is implemented for future compatibility to allow very large amounts of memory storage.

This basic design will allow the method of Direct Addressing to access the full range of data. There are 4 bits of the opcode required for selecting the instruction (4 bits means up to 16 instructions can be supported but only 15 are used) and 2 bits required for addressing mode (total of 4), making the opcode 6 bits long. This leaves 26 out of the 32 bits available for the operand. If the operand contains 2 addresses for data manipulation then each address can be up to 13 bits long, so the maximum memory the design can implement is 512MB (the 32-bit data bus limitation is 4GB if the Base+Index addressing mode is to be used).

INSTRUCTION FORMAT

Instruction Format is to be of fixed length, this allows quick execution of instructions at the cost of memory space. Since this design is for a specific purpose, the memory space is not a major issue, but battery life and efficiency are important. If the program is executed efficiently, the CPU clock speed can be reduced and therefore battery life will be increased.

REGISTERS

There are 32 GPRs available in this design, which is common in 32-bit RISC processors. Each GPR needs only 5 bits to be addressed which leaves sufficient space in the opcode for 3 addresses to be specified in the register direct addressing mode. This allows an operation to be carried out on two sets of data, while allowing them to remain unchanged and then store the result in a different location. There are no more registers, as there is no need for more. 16 GPRs was considered but 32 were chosen to allow a degree of flexibility in the software.

One register is reserved for storing the state of the player. There are 7 states, therefore 3 bits are needed to specify which state the player is in. The track name of an mp3 file is coded into the file. Because of this, a small buffer is implemented to transfer the track name, character by character. Each character is stored in extended ASCII (UTF-8) format, which uses 8 bits to specify which 1 of 256 possible characters, is used. The buffer transfers each character, then clears itself for the next character. This takes up a further 8 bits in the register. This same register will also store the number of the track currently playing. The maximum number of tracks supported will be limited to 256. This takes up 8 bits in the register. The remaining 13 bits will be used to store the amount of time the track has been playing for. These 13 bits limit the maximum track length to 2hours 6minutes and 32seconds.

The instruction Register contains the next instruction to be executed and the Address Register contains the next memory addresses the instruction will require.

The Status Register is a Special Function Register (SFR). Its contents are updated automatically after every instruction cycle. It contains the bits Z, V, C, and N (Zero, overflow, Carry and Negative). These are flags to indicate the nature of the instruction last instruction executed. These flags are heavily used by the ALU, e.g. for addition with carry, but also used for jump or branch instructions.

BRANCH INSTRUCTIONS

Supported branch instructions in hardware are:

- Jump if Zero, i.e. jump if Z=1
- Jump if less than, i.e. jump if Z+N`V+NV`=1
- Unconditional Branch, i.e. jump always
- CALL, saves value of CP and SP

All other branch instructions will be executed by the microprogram. The CALL instruction was included and set to be the only branch instruction to save CP and SP. As all other branch instructions don't save the contents of SP and CP, the inclusion of this instruction saves memory overhead.

INTERRUPTS

There are four interrupt lines implemented within the CPU, two of which are special interrupt lines. The interrupt with highest priority, IRQ1, is an unmaskable interrupt – it is used for emergency power down, e.g. if the battery is taken out whilst the player is still operating. IRQ4 is used for the General Purpose Timer. This ensures decoding of mp3 is carried out at the appropriate frequency maintaining the quality of the original recording. IRQ2 has a high priority and is used for all functions of the mp3 player apart from the PLAY function; these functions are STOP, FF, REW, SKIP FORWARD, SKIP BACK, and PAUSE. This interrupt can be interrupted by IRQ1 only. Execution of this interrupt must finish before a lower priority interrupt can execute. This interrupt line is shared for a variety of functions, meaning the CPU must complete several cycles to identify which function is operating the line. This will cause the mp3 player to pause the decoding process. This is not a problem, as all the functions on this line have been specifically chosen. None of these functions require continuous playing of the mp3 file.

IRQ3 has only one function assigned, which is the PLAY function.

The LCD co-processor operates the backlight on the LCD screen.

INPUT OUTPUT AND OPERATING SYSTEM

Input/Output is controlled by several interfaces. The control bus ensures correct timing of these interfaces, which will control data flow between modules.

The operating system has been chosen to be incompatible with commercially available ones, as this would require hardware to support their functions (in the form of instructions and addressing modes). These operating systems are also designed for general-purpose applications, but this design is for one specific application – playing mp3 files and transferring them across from another computer. This limitation makes the design of the hardware easier. An efficient operating system can be developed for the designs specific needs. The operating system will make sure that limitations are not superseded (these include the maximum number of tracks and the maximum track length).

FURTHER STUDY

Additional memory is available using the expansion slot located within the unit. This allows up to 512MB of memory to be used in the player (equating to approximately 10.5 hours of music at 112kbps).

Power will be supplied to player using Lithium Polymer Battery (LPB) Technology. This new technology offers outstanding battery life and virtually no memory effects by recharging incorrectly. The RISC chip is designed for a handheld portable unit, the rechargeable battery alone will be unable to supply a constant voltage to the chip, which may cause a lot of problems. The solution for this is to use a power supply controller. In the basic form, a step-down chopper may be used, however this doesn't allow variable voltage level regulation. A form of Pulse Width Modulation (PWM) inverter will allow variable voltage output. PWMs have variable duty-cycle allowing the output voltage level to be changed. This is a whole area of study, therefore a chip will be chosen from a manufacturer for this. The one chosen will be a LM2640 from National Semiconductors. This chip supplies constant 2.2V to the RISC chip. This chip may also be used for recharging the battery. This will allow rapid charging rather than trickle charging, often associated with rechargeable batteries.

There are methods of improving the efficiency of the CPU. Time has not allowed these methods to be studied to see if they would be advantageous or not. These include:

- Pipelining – This is when the CPU carries out multiple operations at once. It has separate modules for:
    - Fetch instruction
    - Decode Instruction
    - Calculate Operands (addresses etc.)
    - Fetch Operands
    - Execute Instruction
    - Write Results back into memory

  This method allows multiple one instruction to be executed almost every cycle (with the exception of branch instructions, whose address relies on previous instruction result)

- Branch Predictions – These are used to predict what instructions are likely to result in a jump. Some instructions are 75% likely to result in a jump therefore the CPU uses methods to implement these predictions into the pipeline process.

These methods require a more complex design of CPU and use more silicon, but they make operation of CPU more efficient. It is not known if this would be significant in respect to this design, but it is an interesting area of study for further project work.

ACCEPTANCE TEST PROCEDURES

These tests can be carried out according to the FSM (Finite State Machine) in the form of a state transition table shown below (fig 3).

State	Button Pressed	State Change
PLAY	PLAY	None
PLAY	STOP	STOP
PLAY	FF	FF
PLAY	REW	REW
PLAY	SKIP FORWARD	SKIP FORWARD
PLAY	SKIP BACK	SKIP BACK
PLAY	PAUSE	PAUSE
STOP	PLAY	PLAY
STOP	STOP	None
STOP	FF	None
STOP	REW	None
STOP	SKIP FORWARD	None
STOP	SKIP BACK	None
STOP	PAUSE	None
FF	PLAY	PLAY
FF	STOP	PLAY
FF	FF	PLAY
FF	REW	PLAY
FF	SKIP FORWARD	SKIP FORWARD
FF	SKIP BACK	SKIP BACK
FF	PAUSE	PAUSE
REW	PLAY	PLAY
REW	STOP	PLAY
REW	FF	PLAY
REW	REW	PLAY
REW	SKIP FORWARD	SKIP FORWARD
REW	SKIP BACK	SKIP BACK
REW	PAUSE	PAUSE
SKIP FORWARD	PLAY	PLAY
SKIP FORWARD	STOP	STOP
SKIP FORWARD	FF	None
SKIP FORWARD	REW	None
SKIP FORWARD	SKIP FORWARD	None
SKIP FORWARD	SKIP BACK	None
SKIP FORWARD	PAUSE	PAUSE

State Transition Table (continued)

State	Button Pressed	State Change
SKIP BACKWARD	PLAY	None
SKIP BACKWARD	STOP	STOP
SKIP BACKWARD	FF	None
SKIP BACKWARD	REW	None
SKIP BACKWARD	SKIP FORWARD	None
SKIP BACKWARD	SKIP BACK	None
SKIP BACKWARD	PAUSE	PAUSE
PAUSE	PLAY	PLAY
PAUSE	STOP	STOP
PAUSE	FF	None
PAUSE	REW	None
PAUSE	SKIP FORWARD	SKIP FORWARD
PAUSE	SKIP BACK	SKIP BACK
PAUSE	PAUSE	PLAY
Any	Emergency Power Down	Emergency Power Down

Figure 3 - ATP State Transition Table

To transfer data over the USB link, the unit must be located in the cradle provided. This will allow the unit to enter CONNECT state. A program running on any connected PC will control data flow and buttons will not function, however emergency power may still operate.

CONCLUSION

The design of the processor was confusing to start with as it was unclear how detailed or specific the CPU had to be. I went into a lot of detail about mp3 files and how they are compressed but did not supply any suggestions of how they would be decoded. Simple instructions were implemented in the design that should be sufficient for the operation of decoding and transferring mp3 files.

Overall, I found the design very interesting, but research was difficult. Many resources have different names for addressing modes and these often overlap making it difficult to know which is the correct mode.

The mp3 technology is fascinating, especially in the techniques used to reduce the size of the files but maintaining the quality of the original recording.

I would like to take this study further in the future, perhaps as my main project for my degree.

APPENDIX – MP3 TECHNOLOGY

Mp3 bit rate

The bit rate of an mp3 file (or indeed any other streaming audio, or video file) is played back with a certain amount of data bits per unit time. For audio applications, the convenient unit to use is kilobits per second (kbps, which is 1024 bits of data per second of playback). Compact disc audio is usually encoded with 1.4Mbps. Most mp3 files can be compressed to a ratio of 1:12 maintaining the high quality of the original. This equates to bit rates of around 112kbps to 128kbps. Most mp3 codecs (COder/DECoders) use a fixed bit rate; in other words, for every second of data, the output will have a fixed number of bits. Newer codecs (for example Xing Technologies` codec) use a technique called Variable Bit Rate. These techniques produce a variable amount of bits for every second of audio depending on the complexity of the sound. To select a quality of output, the user specifies an index and the codec produces data rates from 32kbps upwards according to this index.

Minimal Audition Threshold

The human ear does not respond to sounds in a linear manner. In optics, the eye can detect green light approximately 100 times better than red light. In acoustics, the ears detect certain frequencies better than others. Hearing is most responsive to frequencies of around the 2kHz to 5kHz range. Within this range, the ear can detect sounds at very low volumes whereas out with this range, a louder sound is needed for the ear to hear it. The graph[1] below (fig 4) shows the relationship of frequency and Minimum volume (in dB) required for the average human ear to detect a sound:

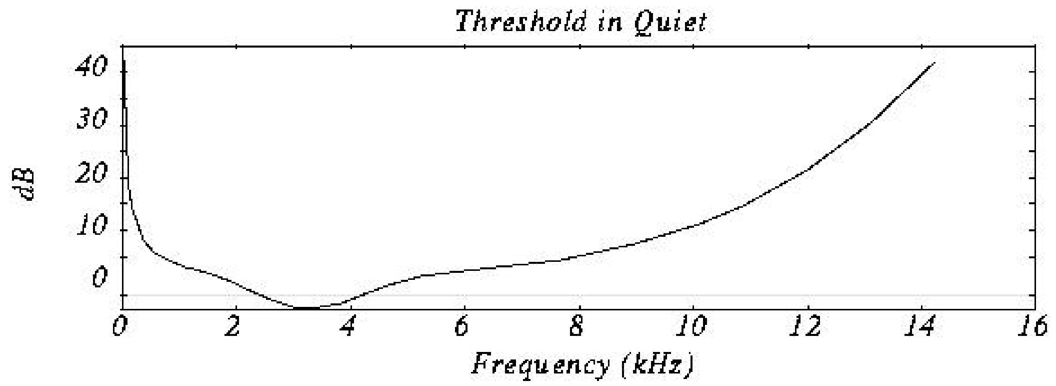

figure 4 - Graph showing Minimal Audition Threshold

This line is known as the Minimal Audition Threshold. The ear does not perceive anything below this line and therefore mp3 files are not coded with anything below it. In optics, there exists the visible spectrum, a range of frequencies that the eye can detect – from approximately 1.4 MHz for red light to 2.5MHz for blue light. In audio, there is a similar phenomenon, the approximate audio range of a good ear is from 20Hz to 20kHz, but these values deteriorate with age. This range sets a cut off point for mp3 files to stop coding i.e. anything out with this range is not coded into the mp3 file.

The Masking Effect

The Masking Effect uses the concept of masking properties. Referring back to optics again, if the eye looks at the sun and a bird flies past, the bird is unseen as the suns light is to predominant. The ear has a very similar effect e.g. with low background volumes, the ear can detect very quiet sounds, like a pin dropping, however if a pin

[1] Graph taken from http://sites.netscape.net/beertent1/main/psycho1.htm

were to be dropped in a rock concert then nobody would here it. This concept is used heavily in is utilised in mp3 encoding. Any sounds that would be masked by louder, more predominant ones are not coded.

Joint Stereo

Joint Stereo is one of many techniques used in coding mp3 files; this one is mentioned, as it is the most popular. There are two main ideas behind Joint Stereo. The first concept, called intensity stereo, is that of spatial position. The human ear cannot accurately locate the spatial origin of a sound with very low or high frequency. Many hi-fi systems contain a sub-woofer to produce very low frequency bass sounds. The location of this sound is undeterminable so the smaller mid-range speakers produce acoustic positioning. Mp3 files use this idea to produce more compressed files. A stereo signal with low or high frequencies is coded to produce a mono signal. The ear cannot tell this, so the mp3 file sounds very close to the original signal.

The second concept of joint stereo is known as Mid/Side stereo. This is used when the left and right channels of a stereo signal are very similar. A middle (L+R) channel and side channels (L-R) are produced instead of separate left and right channels. This allows the side channels to be coded at lower bit rates, as there is less information that is uncommon to both channels. This means smaller files that maintain the high quality of the original recording are produced. During playback of mp3 files, these two channels are output as standard left and right audio channels.

Huffman Coding

Huffman Coding is a coding algorithm used to compress many types of files. It uses unique codes for individual sequences of bits; sequences with high probability are given short codes. This type of compression is simple and decoding can be implemented very quickly making it ideal for real time applications. Typical compression of files is around 20% less than the original. This figure can be increased if no masking effect techniques are implemented as bit sequences are repeated a lot. If masking effects are taken into consideration, the amount of repeated bits is less therefore Huffman compression is less.

Good codecs have a good balance of these techniques to give smaller files that maintain the high quality of the original recording.

BIBLIOGRAPHY

1. EG40GA Course Notes and Handouts – Tim Spracklen, Aberdeen University 2000
2. The Central Processing Unit – unknown (handed out in lecture course EG40GA)
3. PR31700 Data Sheet – Philips Semiconductors, 1998
4. VS1001 Data Sheet – VLSI solution, 2000 (http://www.vlsi.fi/vs1001.html)
5. http://www.iocon.com/das/technical_enc.shtml
6. http://www.mp3-tech.org/tech.html
7. http://iis.fhg.de/amm/techinf/layer3/index.html
8. http://www.newi.ac.uk/pullina/CSD/Computer_Systems_Design_scheme.htm
9. http://sites.netscape.net/beertent1/main/psycho1.htm
10. http://www-personal.umich.edu/~jashbury/audioonline/whatismp3.html
11. http://www.macobserver.com/newreviews/bc/00/000225n2mp3/n2mp3.html
12. http://www.national.com/pf/LM/LM2640.html