介紹如何移植 bento4 到 Andes 平臺上

隨著當今市場對於音訊電子產品功能的需求越來越高，8 位和 16 位的 MCU 逐漸向 32 位的 MCU 轉型已經成爲市場趨勢。晶心科技(Andes)作爲亞洲首家原創性 32 位元微處理器 IP 與系統晶片開發平臺的設計公司，面向 32 位 MCU 市場推出了 Andes Core N9, N10, N12 三个系列的低功耗高性價比的 32 位處理器軟硬核 IP。基於各个系列處理器,晶心科技針對不同音訊應用提供了多種解決方案。包括各種音訊格式的編碼和解碼，如 AAC，MP3，MP4，G729 等移植到 Andes 平臺上。再加上 nds32 架構的優勢和音訊的擴展指令集，以及演算法上的改進，所以編碼解碼器有了進一步的優化，使其代碼空間變小，運行性能變高。本文以 MP4 的音訊解碼器爲例，介紹如何 bento4 移植到 Andes 平臺上。

1.MP4 簡介:

MP4 爲 MPEG-4 Part14，按照正規的 ISO/IEC 14496-14:2003 標準，其作爲 MPEG－4 的一部分是一個多媒體存儲格式專用的標準。一般常常用來存放數位音訊和數位視訊串流檔(音訊可以是：AAC， MP3，MP2 等,視頻可以是 H.264，MPEG-4 AVC 等)，也可以存放其他特殊的資料像字幕和靜止圖像。更很多現在的儲存格式一樣 MPEG－4 Part14 允許在網際網路上傳輸.這些檔的尾碼名是.mp4。

MPEG－4 Part14 是基於蘋果公司的 Quicktime 儲存格式來實現。MPEG－4 Part14 是基本上和 QuickTime MOV 格式相同，但是它有專門支持 Initial Object Descriptors(IOD)格式和其他的 MPEG 特徵。

下面我就詳細的介紹下 MP4 怎樣在我們 Andes 提供的 AndeSight 集成開發工具上解碼。

2.環境與軟體:

2.1 系統環境:

Linux Fedora8

2.2 開發環境:

AndeSight1.4

AndeSight 是晶心科技提供的一種基於 nds32 架構開發嵌入式工程的圖形化的整合式開發環境。主要由 AndeSight IDE, AndESLive 和 nds32 工具集 3 個部分組成。

AndeSight IDE 爲工程師提供了各種友好的介面，包括編輯，編譯，運行，除錯或者評測等操作。

AndESLive 提供了基於 nds32 架構的模擬器和一種圖形化的虛擬 SoC 構建模型，它與 AndeSight IDE 相結合爲使用者提供了一個虛擬的硬體平臺。這個虛擬評估平臺提供 Andes 自行定義 ISA 的多組系列 32 位 CPU IP 以及各種週邊設備 IP，並且支援用戶自訂 IP 模型。

AndESLive 配合 AndeSight IDE 不僅使得 SoC 設計者能在計畫初期就開始軟體設計、偵錯、最優化等工作，並對系統架構及功能進行檢驗，而且使硬體工程師和軟體工程師具有一樣的能力去製作和修改他們各自的系統模型，可以有效的控制 NRE(NonRecurring Engineering)成本，讓軟體工程師在拿到硬體原型之前，即可以進行軟體的開發和優化。

2.3 交叉編譯器:

nds32le-linux-gcc

nds32 工具集中對應不同的 Andes Core 型號，不同的系統函式程式庫以及大小端形式等條件，提供了各種對應的交叉編譯器。這裡我們選用 nds32le-linux-gcc。

2.4 套裝軟體:

首先要從 http://sourceforge.net/projects/bento4

和 http://sourceforge.net/projects/melo 下載 Bento4-0.9.4.002 和 Melo-1.0.0.tar.gz 這兩個

包，把他們存放到指定目錄下。

3 Bento4 的移植

3.1 編譯工程

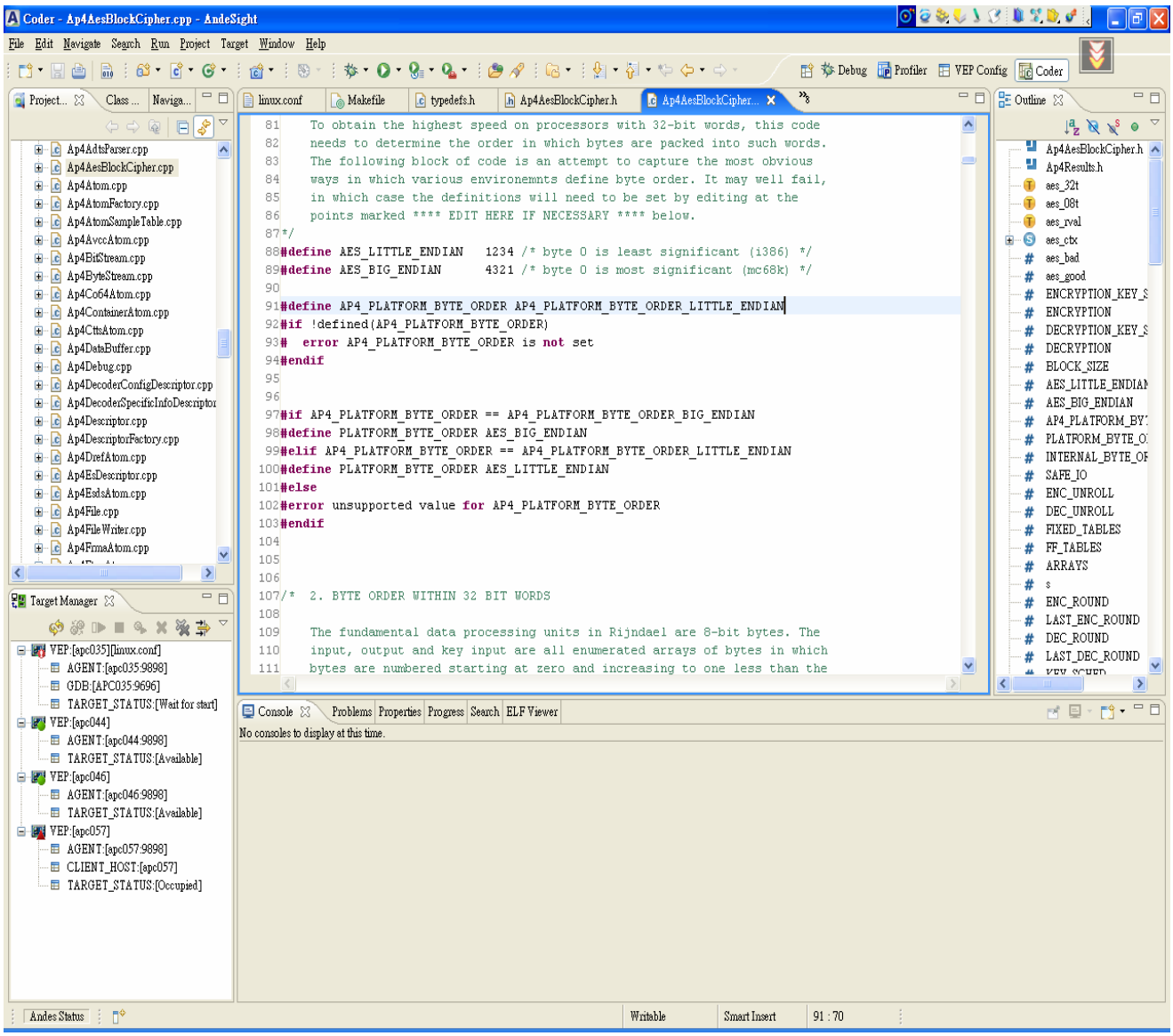
操作的步驟：

第一步：先將下載下來的兩個檔案夾解壓到一個指定的目錄。

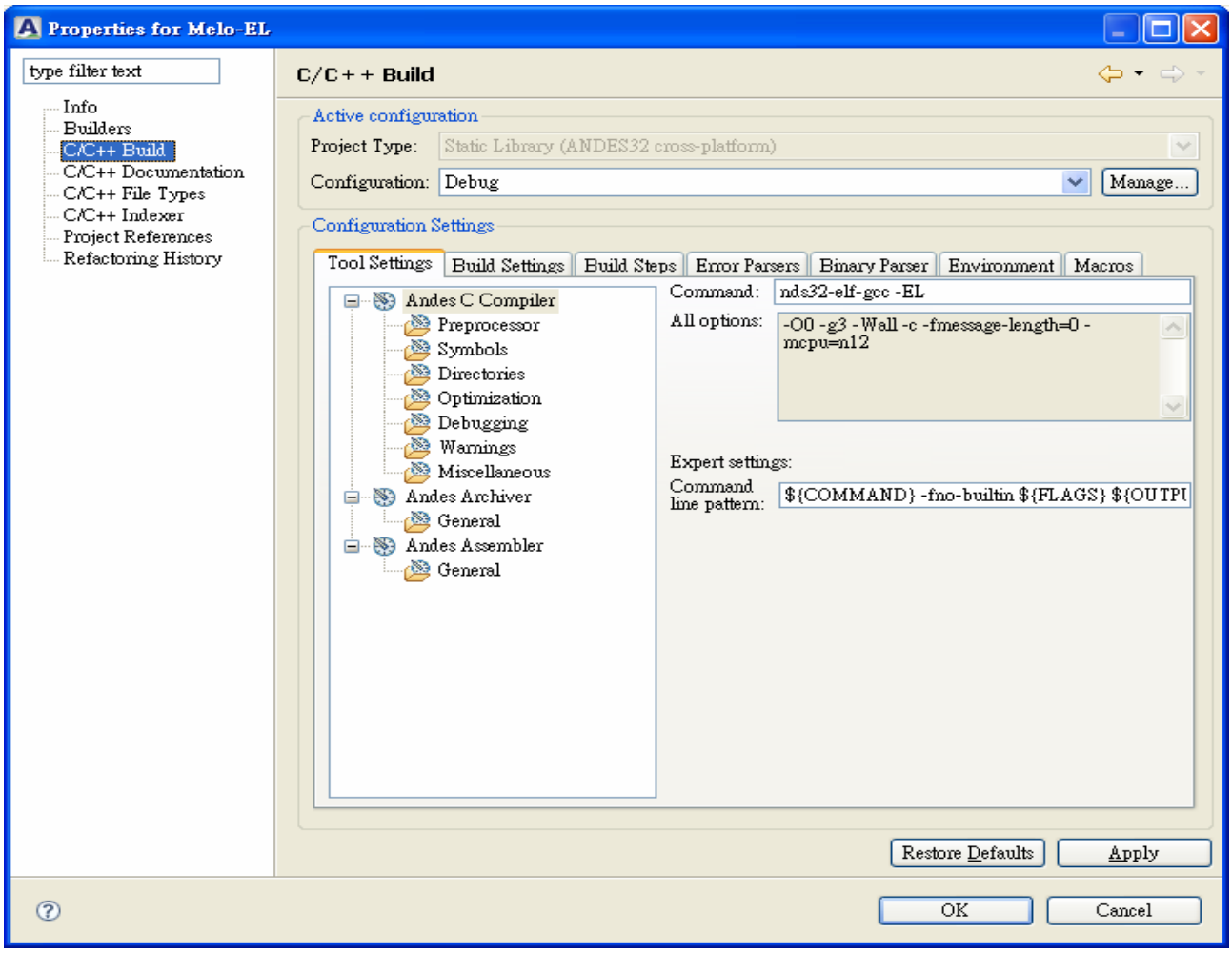
第二步：運行 AndeSight1.3，創建一個 Andes C++工程，命名爲 Bento4-EL。

第三步：在 Bento4-EL 工程中導入所有原始程式碼，找到剛才解壓出來的文檔目錄 Bento4-0.9.4.002,到該目錄下的 Source/C++/，將其中的 Core，Codecs，Metadata，Crypto 和 System/下的 StdC 導入 Bento4-EL 工程。

第四步：找到 Ap4AesBlockCipher.cpp 文件，在該文件的第 90 行加入一句： #define AP4_PLATFORM_BYTE_ORDER AP4_PLATFORM_BYTE_ORDER_LITTLE_ENDIAN。



第五步：先選擇一款 Toolchain，然後編譯 Bento4-EL 工程指定編譯選項-static 和 -EL,AndeSight1.33 默認就是-static 和-EL 編譯,所以不需要改動。



第六步：新建一個名爲 Melo-EL 的 C 工程的靜態程式庫，庫的名字命名爲 melo.a，找到 Melo-1.0.0 目錄，導入 Melo-1.0.0/Source 下所有的檔,然後用小端編譯。

New Project

Select a type of project

Select the platform and configurations you wish to deploy on

Andes

Project Type: Static Library (ANDES32 cross-platform)

Configurations:

- [x] Debug
- [x] Release

Select All

Deselect All

- [ ] Show All Project Types
- [ ] Show All Configurations

< Back | Next > | Finish | Cancel

Properties for Melo_2

type filter text

- Info
- Builders
- C/C++ Build
- C/C++ Documenta
- C/C++ File Types
- C/C++ Indexer
- Project References
- Refactoring History

C/C++ Build

Active configuration

Project Type: Static Library (ANDES32 cross-platform)

Configuration: Debug | Manage...

Configuration Settings

Tool Settings | Toolchain | Build Settings | Build Steps | Error Parsers | Binary Parser | Environment | Macros

Build output

Artifact name: melo . Artifact extension: a

Build command

- [x] Use default command

make -k

Build Macros usage

- [ ] Expand Build Environment Macros

Internal Builder

NOTE: This is experimental functionality

- [ ] Enable Internal Builder
- [x] Ignore build errors

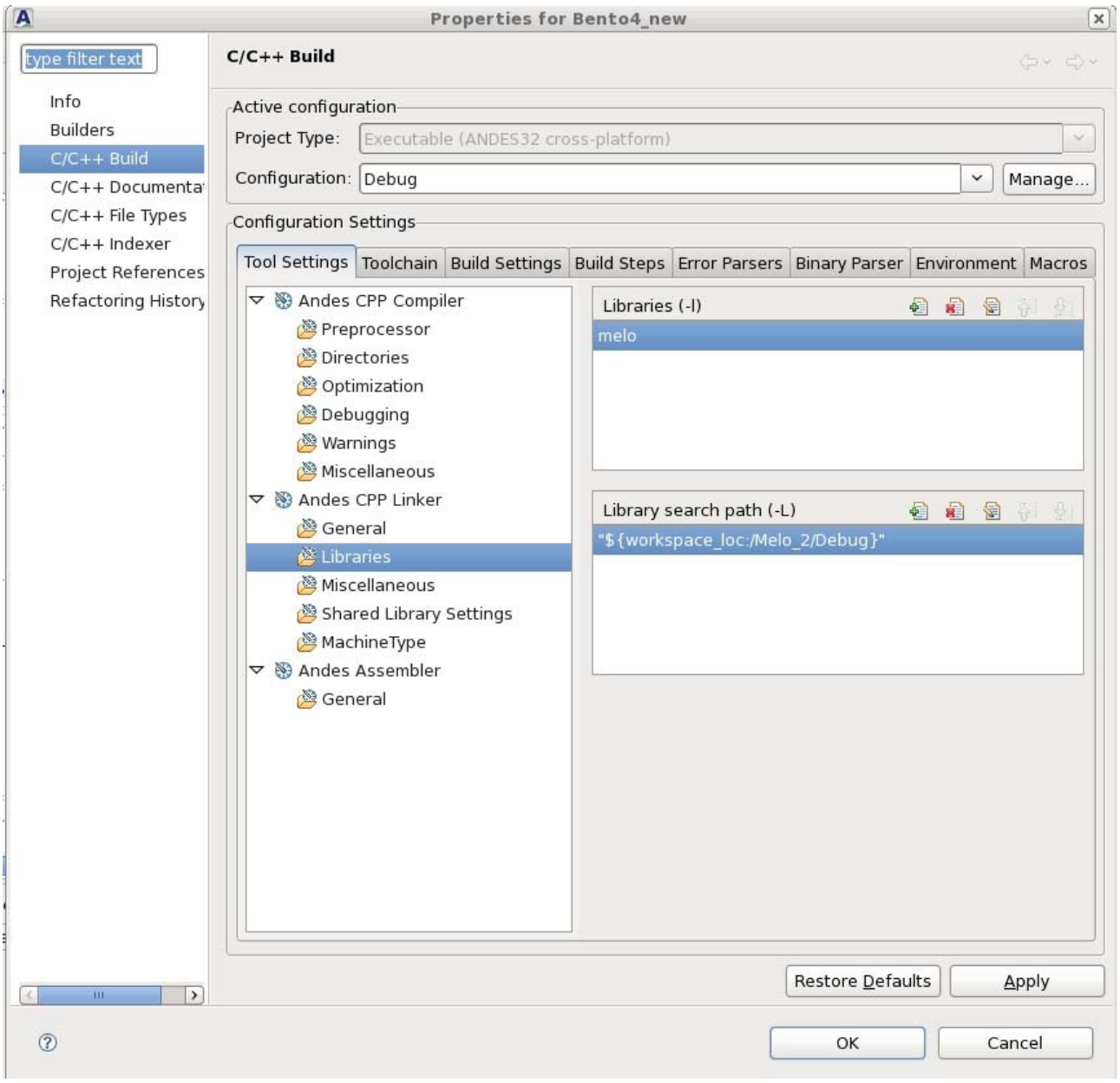
Restore Defaults | Apply

OK | Cancel

第七步：導入 Melo-1.0.0/Apps/MeloDecode 中的 MeloDecoder.cpp 文件，然後把第 128 行的"MLO_SampleBuffer* pcm_buffer = NULL;"搬到第 121 行，然後保存。

```
112     MLO_DecoderConfig decoder_config;
113     MLO_Result result = MLO_DecoderConfig_Parse(encoded_config->GetData(),
114                                                 encoded_config->GetDataSize(),
115                                                 &decoder_config);
116     if (MLO_FAILED(result)) {
117         fprintf(stderr, "ERROR: decoder config is unsupported or unsupported (%d)\n", result);
118         return;
119 }
120
121     MLO_SampleBuffer* pcm_buffer = NULL;
122     // create the decoder
123     MLO_Decoder* decoder = NULL;
124     result = MLO_Decoder_Create(&decoder_config, &decoder);
125     if (MLO_FAILED(result)) {
126         fprintf(stderr, "ERROR: failed to created MLO_Decoder (%d)\n", result);
127         goto end;
128 }
129
130     result = MLO_SampleBuffer_Create(0, &pcm_buffer);
131     if (MLO_FAILED(result)) goto end;
132
133     while (AP4_SUCCEEDED(track->ReadSample(index, sample, data))) {
134         result = MLO_Decoder_DecodeFrame(decoder, data.GetData(), data.GetDataSize(), pcm_buffer);
135         printf("MLO_Decoder_DecodeFrame return %d\n", result);
136         output->Write(MLO_SampleBuffer_GetSamples(pcm_buffer), MLO_SampleBuffer_GetSize(pcm_buffer));
137       index++;
138 }
139
140end:
141     if (pcm_buffer) MLO_SampleBuffer_Destroy(pcm_buffer);
142     if (decoder) MLO_Decoder_Destroy(decoder);
143}
```

第八步：把所有 Melo/Source 下的.h 檔導入到工程 Bento4 中，並且把 MeloDecoder.cpp 一併導入，並且在 Andes CPP Linker 的 Libraries 指定-lmelo 和-L path，libraryd 的路徑像："${workspace_loc:/Melo_2/Debug}" ，然後選擇一款 Toolchain, 再編譯(-EL)。

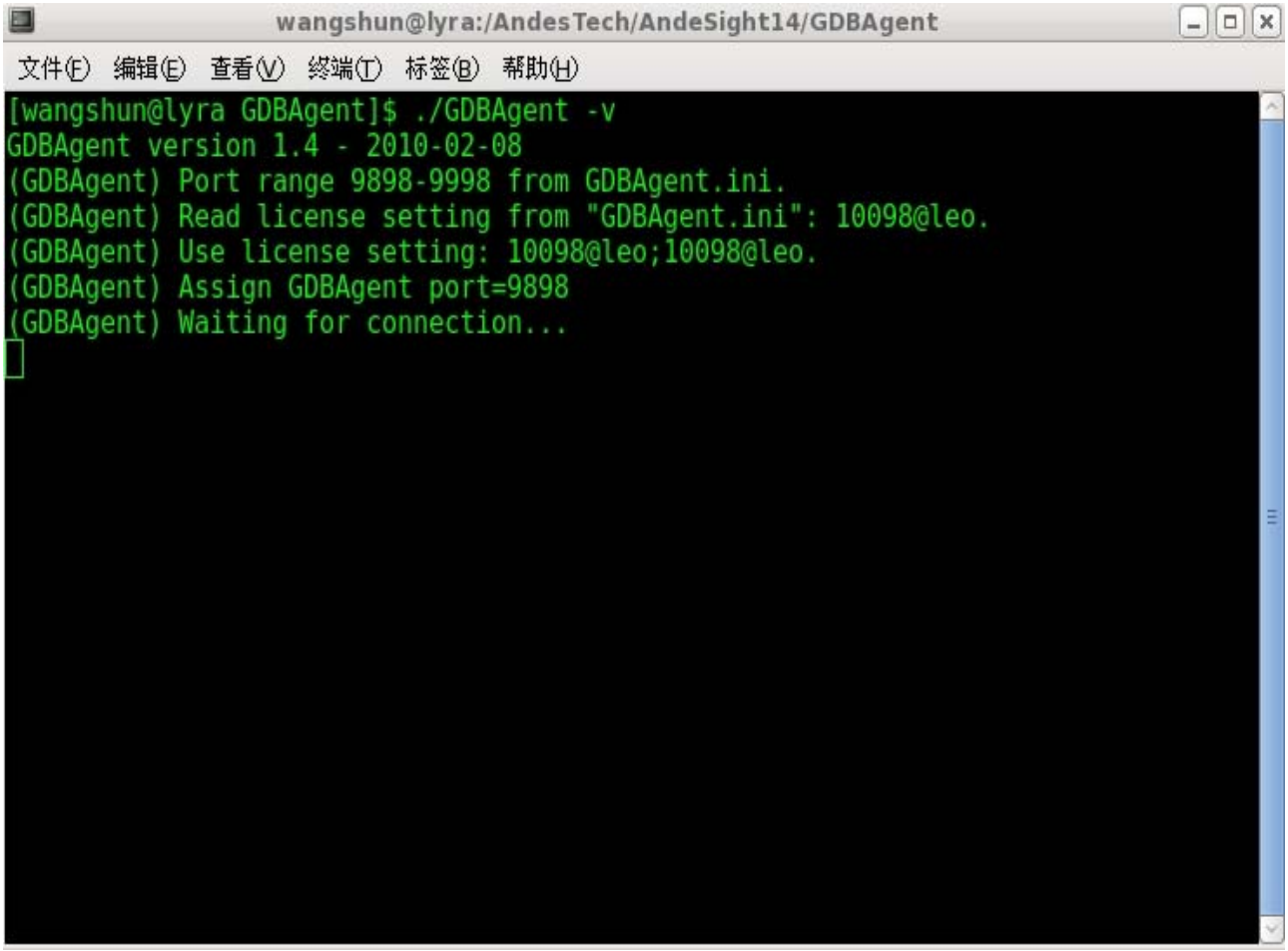

Properties for Bento4_new
type filter text
Info
Builders
C/C++ Build
C/C++ Documenta
C/C++ File Types
C/C++ Indexer
Project References
Refactoring History
C/C++ Build
Active configuration
Project Type:
Executable (ANDES32 cross-platform)
Configuration:
Debug
Manage...
Configuration Settings
Tool Settings
Toolchain
Build Settings
Build Steps
Error Parsers
Binary Parser
Environment
Macros
Andes CPP Compiler
Preprocessor
Directories
Optimization
Debugging
Warnings
Miscellaneous
Andes CPP Linker
General
Libraries
Miscellaneous
Shared Library Settings
MachineType
Andes Assembler
General
Libraries (-l)
melo
Library search path (-L)
"${workspace_loc:/Melo_2/Debug}"
Restore Defaults
Apply
OK
Cancel

3.2 運行工程

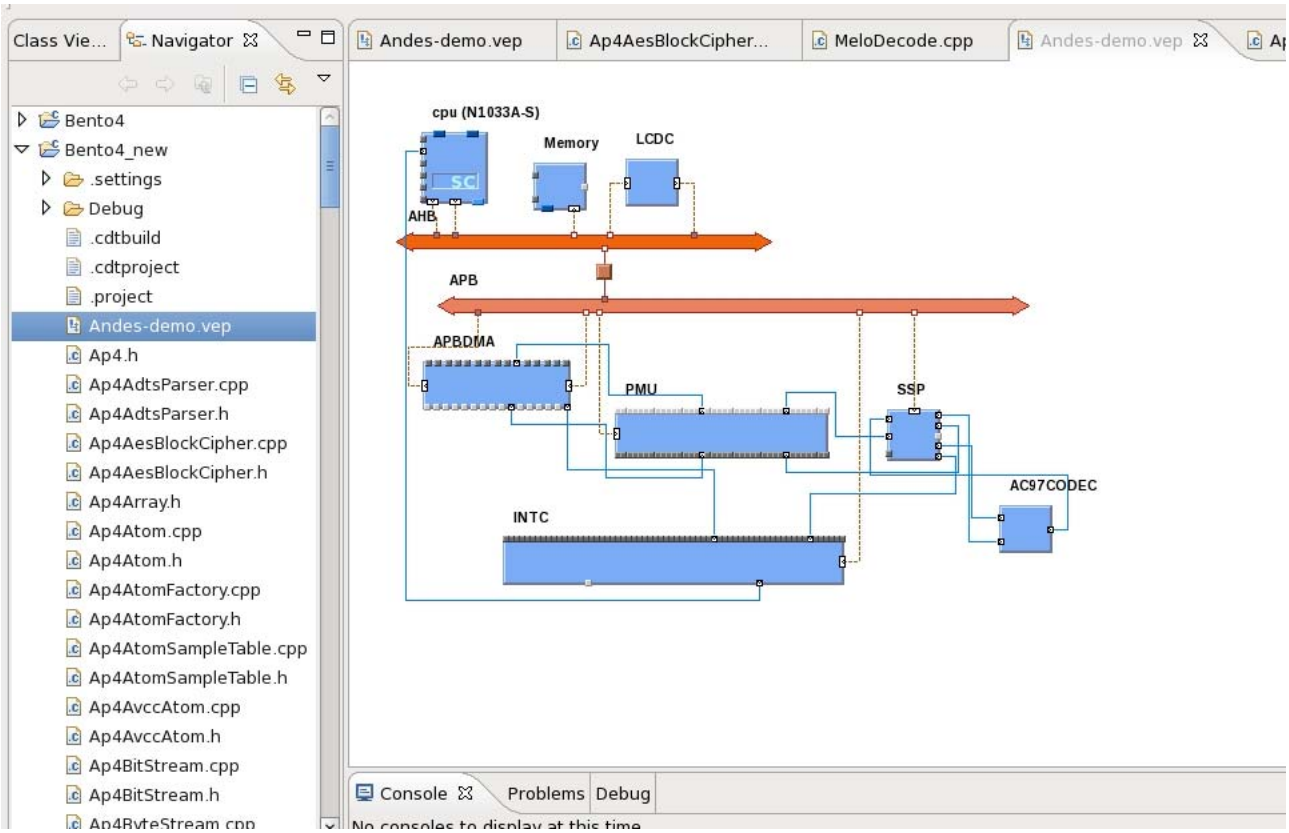
以上這些步驟就是所有的編譯過程，最終會成長一個 MP4 decoder 的可執行檔。

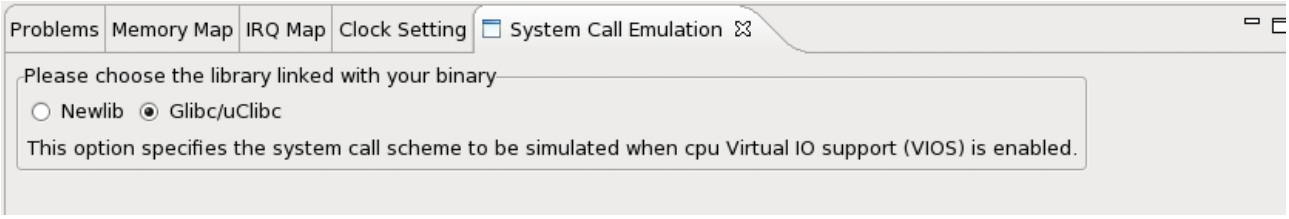
接下來是運行的部分：

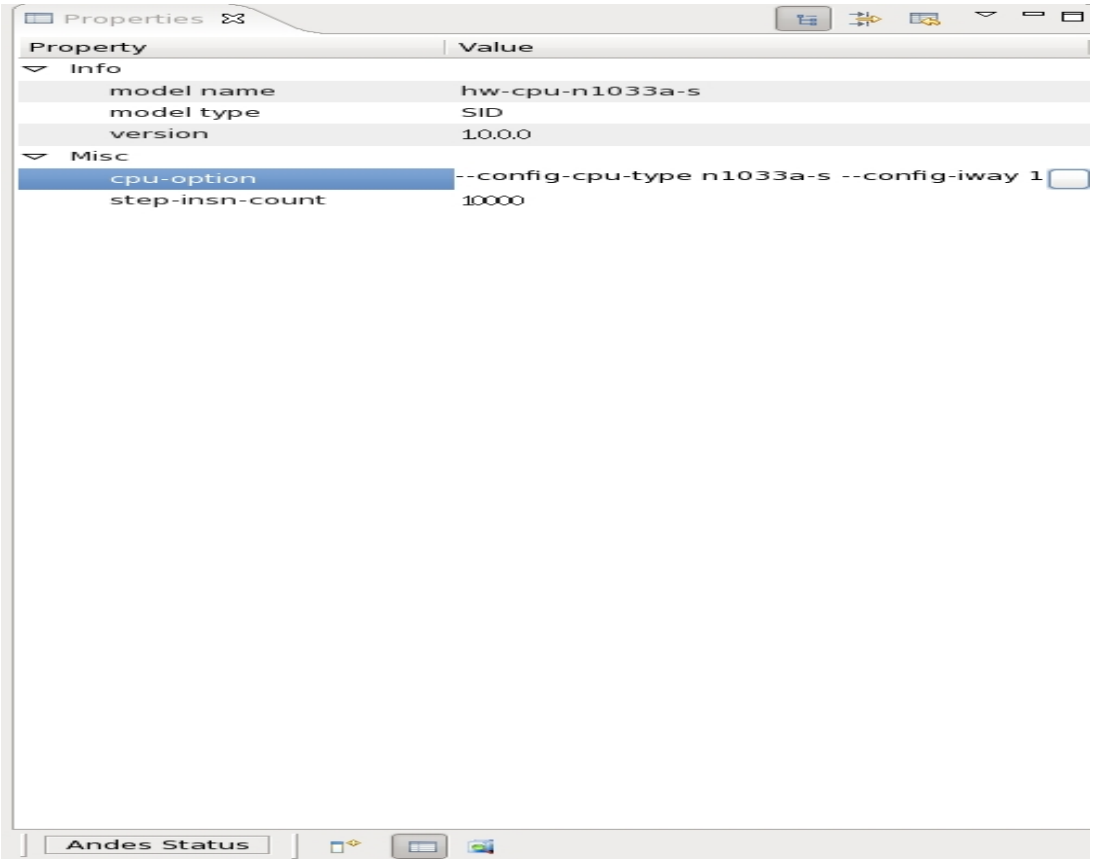
第一步：說到運行首先要開啓我們的 GDBAgent，在 command line 中進入 GDBAgent 這個目錄，在 command line 上輸入：“./GDBAgent -v”，開啓 GDBAgent。

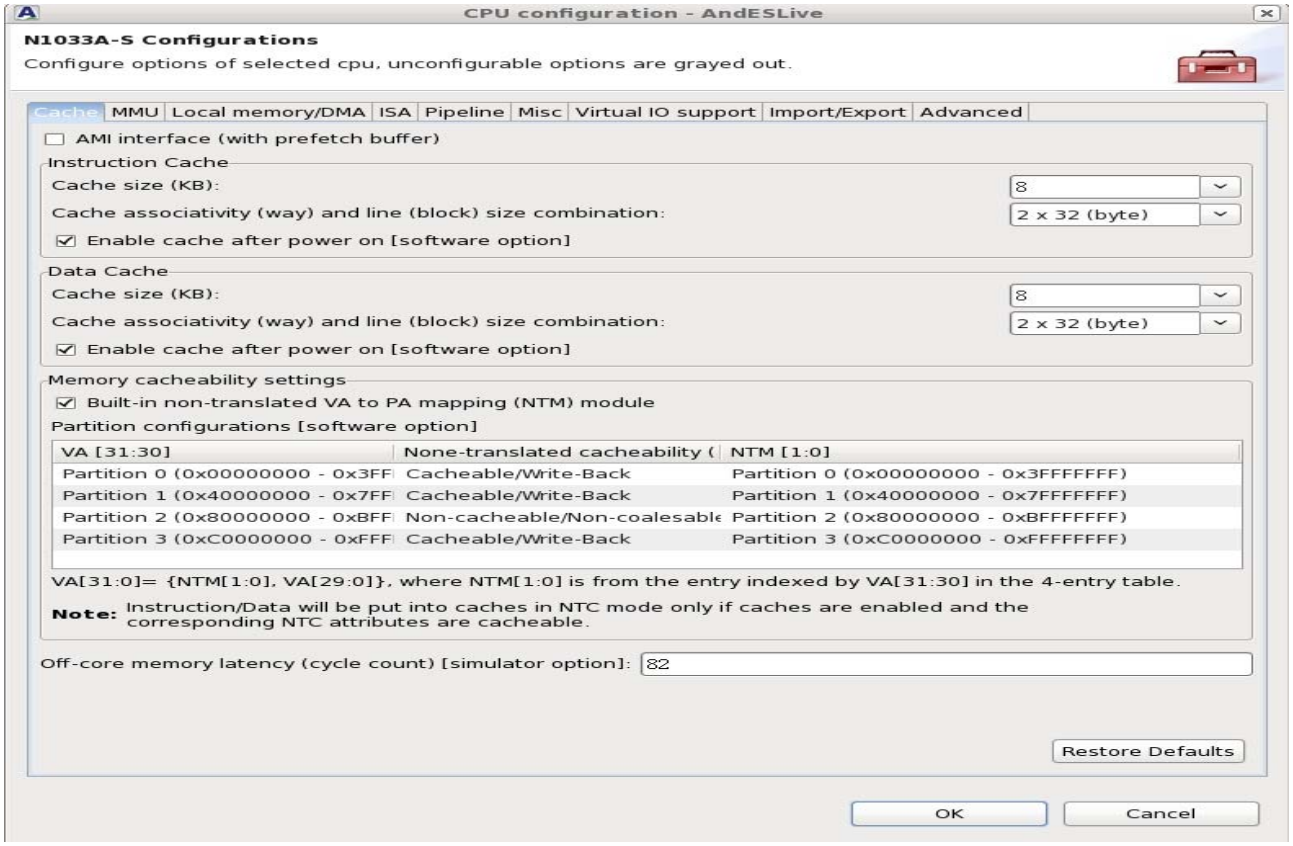


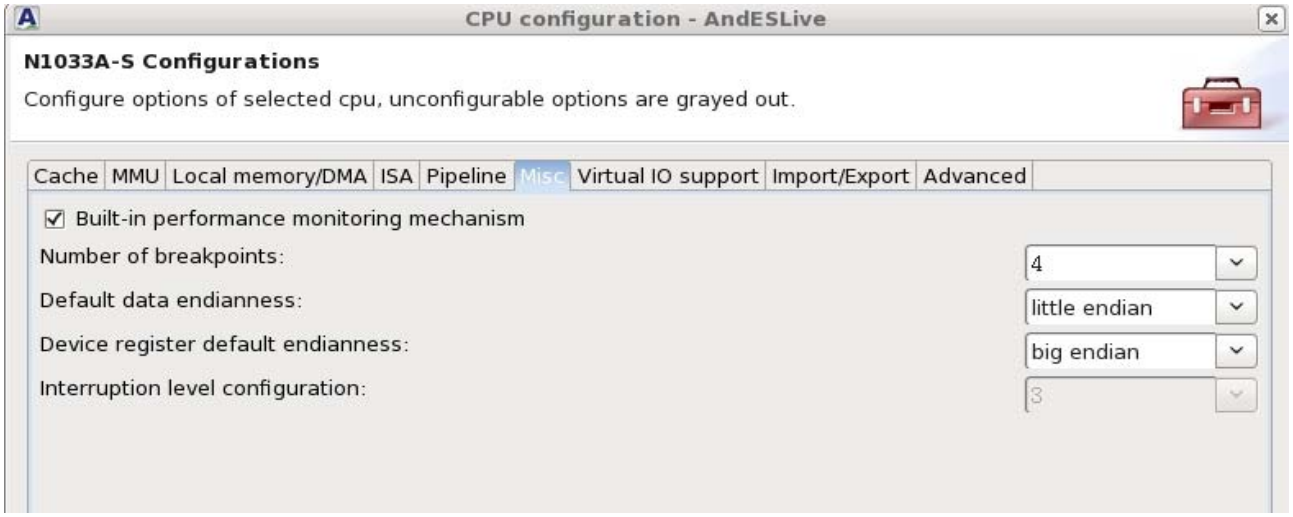
第二步：導入和上面的 Toolchain 相匹配的.vep 文件，如 Andes-demo.vep(一定要可以匹配的)，然後按兩下它,編輯這個檔，設置 cpu 選項,設置 Data endianness 爲 little endian，然後到 vep config 模式下修改 System Call Emulation 的 link library 爲 Glibc。


Class Vie...
Navigator
Andes-demo.vep
Ap4AesBlockCipher...
MeloDecode.cpp
Andes-demo.vep
Bento4
Bento4_new
.settings
Debug
.cdtbuild
.cdtproject
.project
Andes-demo.vep
Ap4.h
Ap4AdtsParser.cpp
Ap4AdtsParser.h
Ap4AesBlockCipher.cpp
Ap4AesBlockCipher.h
Ap4Array.h
Ap4Atom.cpp
Ap4Atom.h
Ap4AtomFactory.cpp
Ap4AtomFactory.h
Ap4AtomSampleTable.cpp
Ap4AtomSampleTable.h
Ap4AvccAtom.cpp
Ap4AvccAtom.h
Ap4BitStream.cpp
Ap4BitStream.h
cpu (N1033A-S)
SC
Memory
LCDC
AHB
APB
APBDMA
PMU
SSP
AC97CODEC
INTC
Console
Problems
Debug
No consoles to display at this time


Problems
Memory Map
IRQ Map
Clock Setting
System Call Emulation
Please choose the library linked with your binary
Newlib
Glibc/uClibc
This option specifies the system call scheme to be simulated when cpu Virtual IO support (VIOS) is enabled.

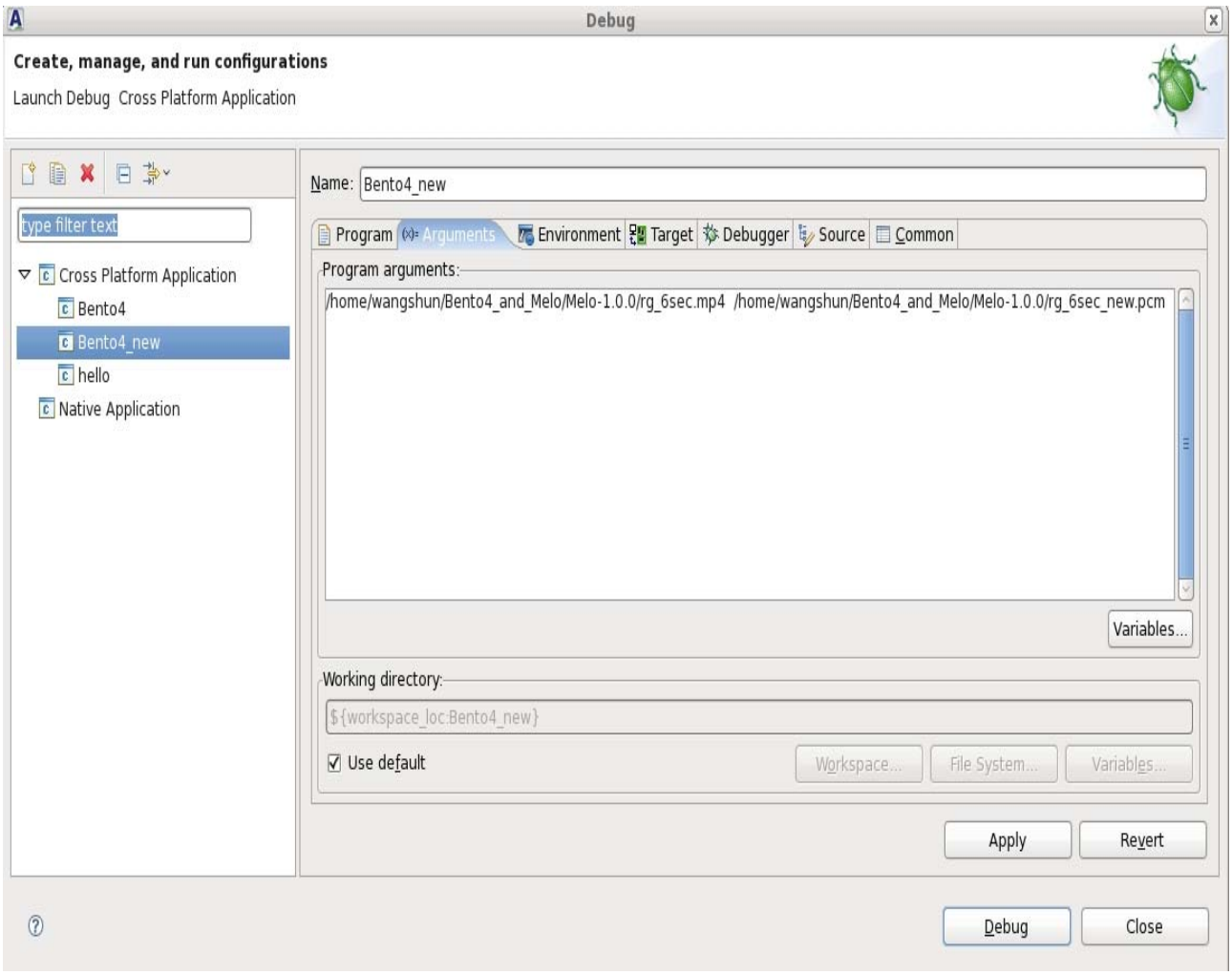

Properties
Property
Value
Info
model name
hw-cpu-n1033a-s
model type
SID
version
1.0.0.0
Misc
cpu-option
--config-cpu-type n1033a-s --config-iway 1
step-insn-count
10000
Andes Status





第三步：找一個 mp4 的檔放到 AndeSight 的工作目錄下，或者下面直接用絕對路徑。

第四步：Debug Bento4-EL 這個工程，在 Debug 對話方塊中，在 Arguments 選項內輸入如：“XXX.mp4 XXX.pcm”，如果你的 mp4 檔不在當前工作目錄下就用絕對路徑。



第五步：然後點擊 Debug 按鈕，你的 mp4 文件就 decoder 成功了，你會看到有個大容量的.pcm 檔在你指定的目錄下，你可以用 mplayer 去運行它，這裡需要設置他的 Sample Rate 爲 44100HZ 和 2 channels，你就會聽到動聽的音樂或清晰的視頻了。

3.3 運行時的 Console

```
Audio Track:
  duration: 233619 ms
  sample count: 10065
    Stream Type: Audio
    Object Type: MPEG-4 Audio
    Max Bitrate: 237888
    Avg Bitrate: 208668
    Buffer Size: 878
    Sample Rate: 44100
    Sample Size: 16
    Channels:    2
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
MLO_Decoder_DecodeFrame return 0
```

3.4 播放 pcm 文档

在 linux 系统安装可以运行 pcm 歌曲的播放器，这里我们使用的是 mplayer，然后使用 mplayer 播放 pcm 歌曲。

```
[wangshun@lyra 04_28]$  mplayer -demuxer rawaudio -rawaudio rate=44100:channe
ls=2 /home/wangshun/Bento4_and_Melo/Melo-1.0.0/rg_6sec_new.pcm
MPlayer 1.0rc2-4.3.2 (C) 2000-2007 MPlayer Team
CPU: Intel(R) Core(TM)2 Duo CPU     E4500  @ 2.20GHz (Family: 6, Model: 15, S
tepping: 13)
CPUflags:  MMX: 1 MMX2: 1 3DNow: 0 3DNow2: 0 SSE: 1 SSE2: 1
Compiled for x86 CPU with extensions: MMX MMX2 SSE SSE2

Playing /home/wangshun/Bento4_and_Melo/Melo-1.0.0/rg_6sec_new.pcm.
rawaudio file format detected.
==========================================================================
Opening audio decoder: [pcm] Uncompressed PCM audio decoder
AUDIO: 44100 Hz, 2 ch, s16le, 1411.2 kbit/100.00% (ratio: 176400->176400)
Selected audio codec: [pcm] afm: pcm (Uncompressed PCM)
==========================================================================
[AO OSS] audio_setup: Can't open audio device /dev/dsp: Device or resource bu
sy
AO: [alsa] 44100Hz 2ch s16le (2 bytes per sample)
Video: no video
Starting playback...
A:   5.6 (05.6) of 6.0 (06.0)  0.1%

Exiting... (End of file)
```

3.5 总结

以上所述方法即可将 Bento4 成功的移植到 andes 平台上。