



Ethernet alapú hanglejátszás DSP-vel

MSc Önálló Laboratórium 2 beszámoló

Gerzsenyi Marcell (MSc)

Konzulens : Orosz György

Méréstechnika és Információs Rendszerek Tanszék

2012 / 13 első félév

Bevezetés

- Motiváció
 - Audió alkalmazással szerettem volna foglalkozni
 - Erre jól alkalmazhatóak a tanszéki DSP kártyák
 - Felmerült az Ethernet használatának lehetősége
- Feladat
 - A DSP kártya Ethernet részének megismerése
 - Hangátviteli lehetőségek vizsgálata
 - Fejlesztői eszközök megismerése

Az előző félév folytatása

- uClinux-ot sikerült elindítani a kártyán
- Ethernet-en, soros porton lehet kommunikálni a kártyával
- Egyszerű alkalmazást már tudtam rá írni

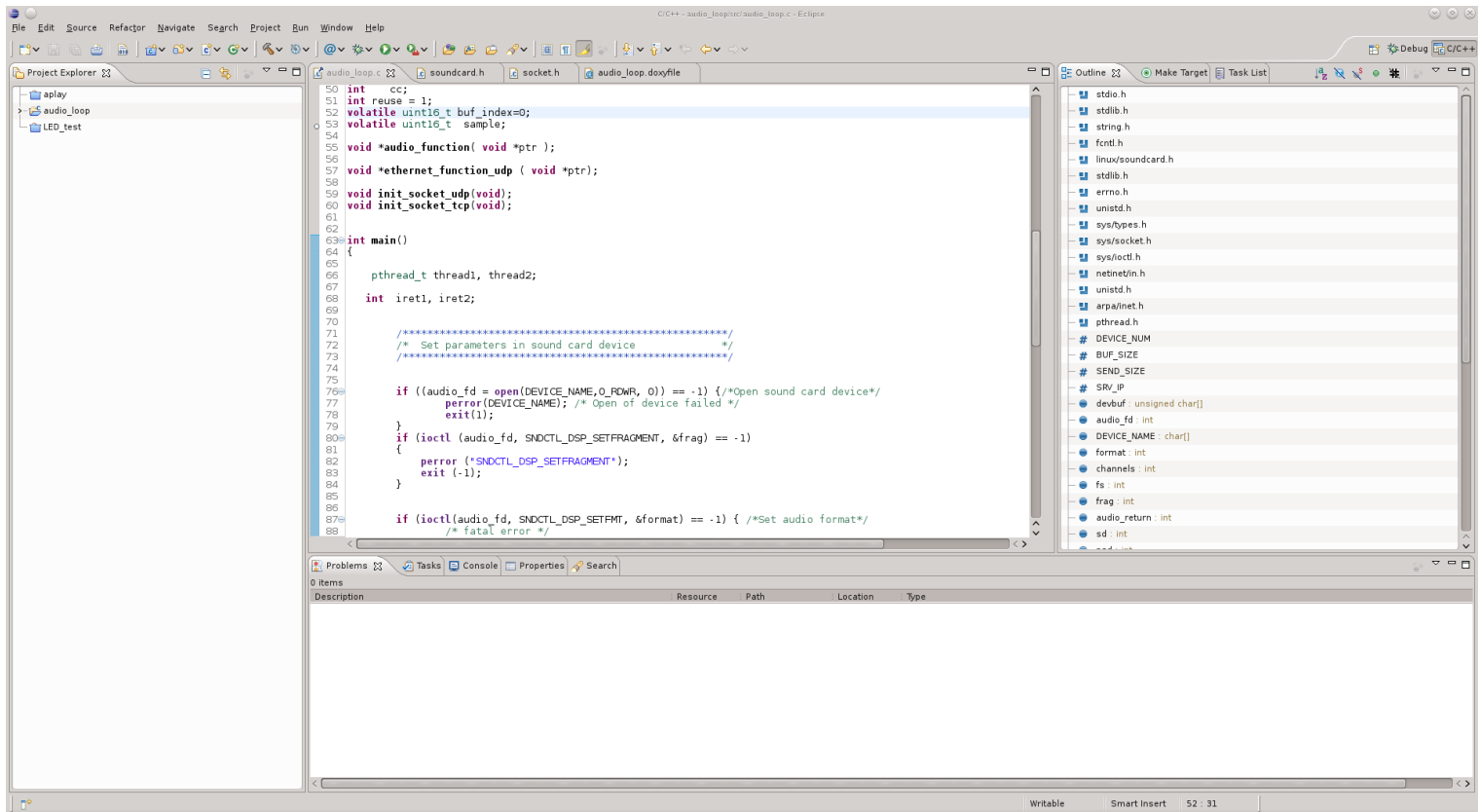
uClinux megismerése

- Közel áll a PC-ken használt linux-hoz
- Direkt mikrokontrollerekre fejlesztve:
 - Linux 2.6 kernel az alapja,
 - De egyszerűsített (Nincs MMU: Memory Management Unit)
- Kész driverek:
 - Ethernet
 - DMA, cache, stb...

Új fejlesztések

- Audio driver alapból nem volt
 - Nyár folyamán kaptam egyet :)
- Kernel optimalizálása
- SD kártyára fájl rendszer telepítése
- Fejlesztőkörnyezet beállítása
 - Eclipse konfiguráció
 - Hasonló a fejlesztés mintha PC-re történne
 - Debuggolás megoldása

A fejlesztőkörnyezet



The screenshot displays the Eclipse IDE interface with a C++ project named 'audio_loop'. The main editor shows the source code for 'audio_loop.c', which includes headers for 'soundcard.h' and 'socket.h'. The code defines several functions: 'audio_function', 'ethernet_function_udp', 'init_socket_udp', and 'init_socket_tcp'. The 'main' function sets up a multi-threaded environment, opens a sound card device, and configures it for audio playback. The IDE also shows a Project Explorer on the left, an Outline view on the right, and a Problems/Console/Properties/Search panel at the bottom.

```
50 int cc;
51 int reuse = 1;
52 volatile uint16_t buf_index=0;
53 volatile uint16_t sample;
54
55 void *audio_function( void *ptr );
56
57 void *ethernet_function_udp ( void *ptr);
58
59 void init_socket_udp(void);
60 void init_socket_tcp(void);
61
62
63 int main()
64 {
65     pthread_t thread1, thread2;
66     int iret1, iret2;
67
68     /******
69     /* Set parameters in sound card device
70     /******
71
72     if ((audio_fd = open(DEVICE_NAME,O_RDWR, 0)) == -1) { /*Open sound card device*/
73         perror(DEVICE_NAME); /* Open of device failed */
74         exit(1);
75     }
76     if (ioctl (audio_fd, SNDCTL_DSP_SETFRAGMENT, &frag) == -1)
77     {
78         perror ("SNDCTL_DSP_SETFRAGMENT");
79         exit (-1);
80     }
81
82     if (ioctl(audio_fd, SNDCTL_DSP_SETFMT, &format) == -1) { /*Set audio format*/
83         /* fatal error */
84     }
```

A tényleges hang átvitel

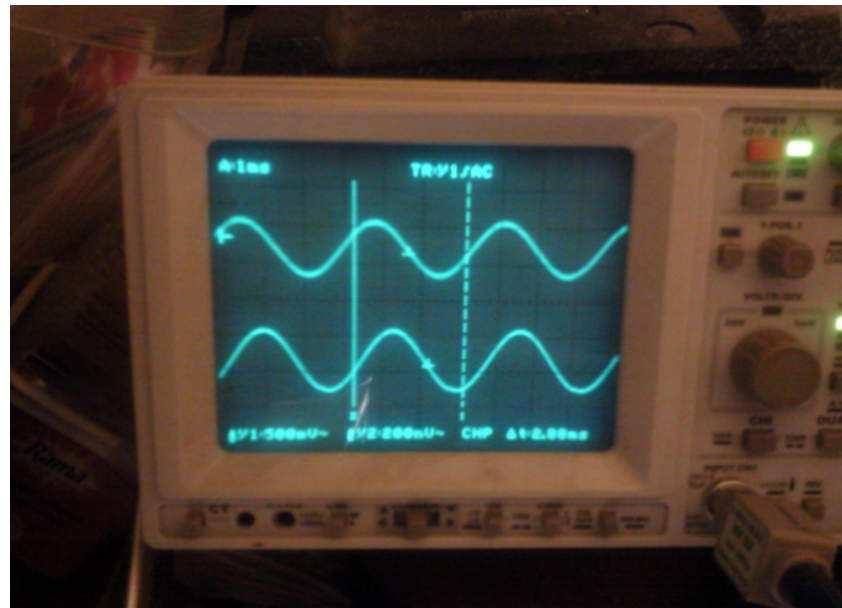
- A fő feladat a megfelelő komponensek integrálása.
- A tervezéskor eldöntendő kérdések
 - Saját megoldás vagy valamilyen szabványos alkalmazása
 - Ethernet átviteli rétege, TCP vagy UDP
 - Hangkártya absztrakciós szintje : OSS vagy ALSA

Az implementáció

- Alapvetően két feladata van a programnak:
 - Hangkártya kezelése
 - Hálózat kezelése
- Megoldás: többszálú program,
Linux alatt Posix thread-del

Elért eredmények

- Sikerült hangot átvinni
 - Egy csatorna átvitele működik
 - Sajnos nagy a késleltetés



Összefoglalás, kitekintés

- Elért célok:
- Ethernet széleskörű használatának felderítése:
 - Op.Rendszer letöltés, felhasználói interface a kártyához, felhasználói program letöltés, debuggolás ethernet keresztül
 - Magas szintű alkalmazások is készen vannak (DHCP kliens, ifconfig stb)
- Audio alkalmazás fejlesztése