Treball de Recerca

openplayer

Construcció i programació d'un reproductor de música portàtil

Autor: Gerard Marull Paretas
Tutor: Jordi Fanals Oriol
2n Batx.Tecnològic

IES La Bisbal
Gener de 2008

Treball realitzat amb OpenOffice.org

Agraïments

A en Qibo Zhang per ajudar-me amb tots els dubtes sobre electrònica
A tota la comunitat d'*AVRFreaks* i l'*AVR-GCC List* per ajudar-me amb dubtes tècnics
A la meva mare per suportar-me durant tot aquest període
Al meu pare per ajudar-me en el procés de construcció
Al tutor i al centre per proporcionar-me alguns dels materials que he necessitat
A alguns altres tecnòlegs del grup que m'han ajudat en la construcció
A aquells professors que hagin tolerat les faltes d'assistència causades per aquest treball
A la meva gata Kitty que va esperar-se al meu costat fins a altes hores de la nit

Finalment a la meva àvia que encara que ja no hi sigui, va donar-me un gran suport moral durant els mesos que va durar aquest treball. Gràcies per tot!

Els dispositius mòbils d'avui dia són més potents que aquells ordinadors amb què treballava només fa 10 anys. No em puc esperar per veure què construiran els innovadors d'avui dia. -Sergei Brin.

Índex

Annexos

Capítol 0

Introducció

1. Origens

Fa un temps, llegint la revista *MAKE* vaig observar que acabaven de publicar un nou controlador anomenat *MAKE Controller*. Aquest contenia un microcontrolador juntament amb una petita placa que disposava d'entrades i sortides, utilitats per controlar motors i fins i tot permetia la connexió a la xarxa. A més, incloïa un petit sistema operatiu per tal de poder començar a programar aplicacions ràpidament i sense massa experiència en aquest camp. La veritat és que hi vaig treballar durant poc temps, bàsicament perquè si volies començar algun codi des de l'inici, és a dir, sense aquest sistema operatiu, era una tasca feixuga ja que la complexitat de l'aparell requeria certa experiència. Així doncs, poques setmanes després vaig decidir cercar algun microcontrolador més simple on poder aprendre el seu funcionament més fàcilment. Resulta que el mateix fabricant del microcontrolador inclòs en el *MAKE Controller* (Atmel) també en fabricava uns altres de més simples: els microcontroladors *AVR*. Vaig cercar exemples d'alguns projectes realitzats amb aquesta sèrie de microcontroladors, i apareixien resultats molt interessants: rellotges, petites estacions meteorològiques, robots, reproductors de música.... Aquest últim em cridava l'atenció, ja que em va semblar *impossible* fer-los amb un microcontrolador de tan baixa potència. Sense dubtar gaire, vaig decidir comprar uns quants *AVR* i vaig començar a fer algunes coses bàsiques amb ells. Uns mesos més tard vaig pensar que el treball de recerca podria estar relacionat amb els microcontroladors i llavors va ser quan vaig fer memòria i vaig retrobar-me amb aquella idea de fer un reproductor de música. Així va néixer l'openplayer.

2. Objectius i consideracions del treball

L'objectiu d'aquest treball sempre ha estat molt clar: la construcció i programació d'un reproductor de música. Tot i això, haig de reconèixer que quan vaig començar tenia una idea molt espessa del que acabaria sent i incloent el reproductor. De fet, mai vaig definir-ne un títol ni objectius fixes, ja que no tenia massa clar com fer certes coses ni tan sols si les podria acabar fent. El procés que més desitjava però que alhora m'atemoria més era el fet de fer-lo portàtil i integrar-lo en una placa de circuit imprès, ja que entregar només un reproductor de sobretaula amb connexions fetes amb fils mal organitzats i d'un tamany que poc s'apropa a qualsevol reproductor comú no té massa sentit. Tot i així, els objectius i la línia de treball han quedat definits de la manera següent:

- **Part teòrica**
    1. Aprendre a utilitzar a fons les possibilitats que ofereix un microcontrolador AVR.
    2. Aprendre l'estructura de treball dels diversos components externs que componen el reproductor.
    3. Extendre els coneixements de programació sota microcontroladors *AVR*.
    4. Aprendre a crear i treballar amb esquemes electrònics i transformar aquests en circuits impresos que a més puguin ser fabricats.
- **Part pràctica**
    1. Programació completa del firmware del reproductor en llenguatge C.
    2. Creació d'un reproductor de sobretaula sobre protoboards on poder treballar i provar el codi que es va programant.
    3. Transformació del reproductor de sobretaula en un de tamany butxaca per tal de fer-lo portàtil.

Finalment, també penso que és important fer esment de l'actitud adoptada davant aquest treball en quant a les llicències que he escollit a l'hora de publicar-lo. Una de les coses que tenia clares és que tant la llicència del codi com la llicència dels documents havien de ser de caràcter lliure, garantint d'aquesta manera que altres usuaris en poguessin fer ús sense restriccions, així que finalment vaig decidir adoptar dues de les llicències que posa a disposició pública la FSF (*Free Software Foundation*). La primera, anomenada GPLv3 (*GNU General Public License version 3*), és la emprada en el codi del firmware del reproductor; la segona, anomenada GFDL (*GNU Free Documentation License*) és la emprada per tots els documents publicats. Podeu obtenir-ne més informació accedint a l'Annex 5, apartat 2.6.

3. Contingut d'aquest documet

La complexitat de l'openplayer ha augmentant considerablement en els últims mesos i per tant escriure en tant poques pàgines una descripció completa i tècnica del reproductor hauria estat impossible. És per aquest motiu que vaig optar per dividir el treball en dues parts: l'actual (aquest document) i una altra d'externa que queda fora del marc d'aquest treball. En aquesta part he decidit tractar el treball d'una manera simple i poc tècnica per tal que sigui el més comprensible possible tot i que potser en algun capítol hagi inclòs alguns termes més tècnics. El treball extern que pretenc publicar en els pròxims mesos (probablement durant l'estiu) a http://teslabs.com/openplayer contindrà una documentació completa del codi juntament amb la descripció tècnica del seu funcionament i de les parts que el componen. Algunes parts d'aquest document es troben accessibles en forma de borrador a l'Annex 5, apartat 2.7.

Capítol 1

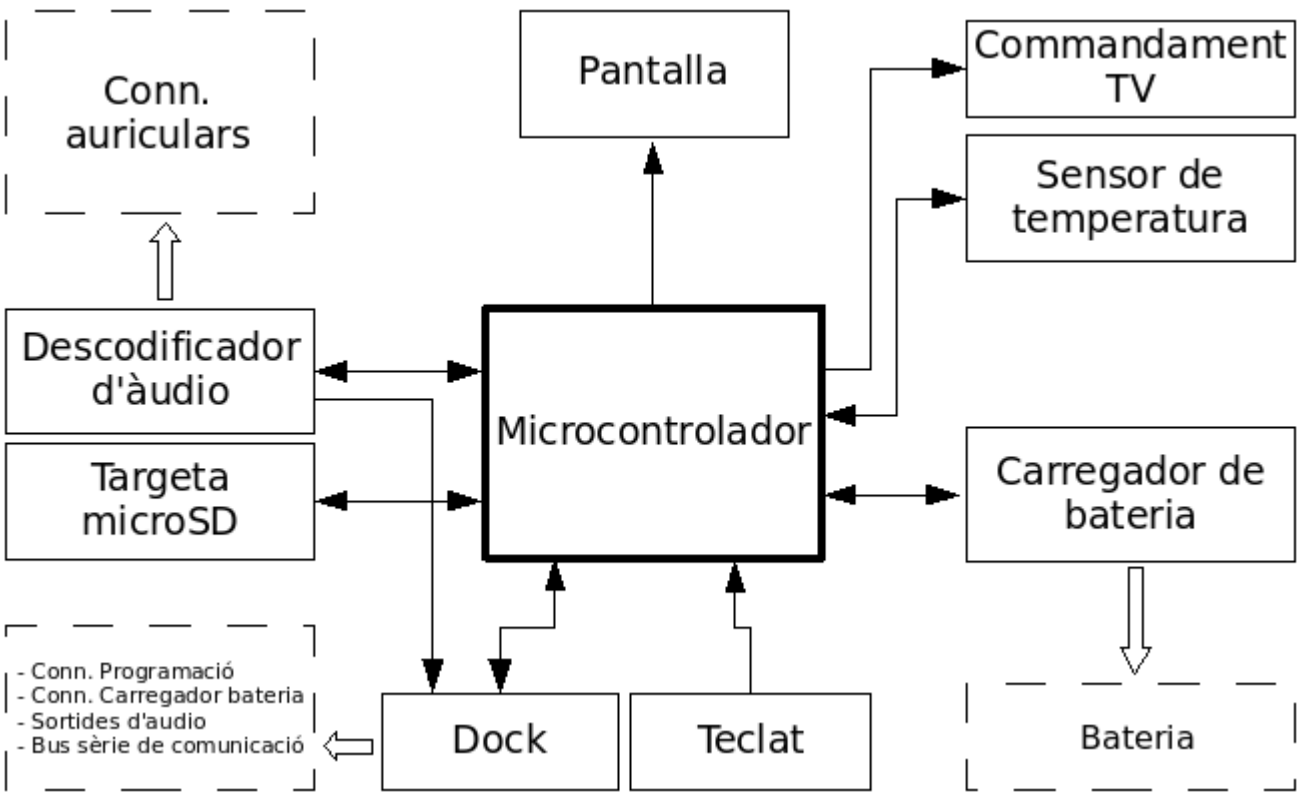
Estructura

1. Els components i la seva relació

L'openplayer consta de diversos components, alguns més rellevants que altres. Possiblement en un reproductor de música comercial aquests components siguin diferents ja que actualment es fabriquen xips on s'integren diversos d'ells en una sola peça o també perquè pot incloure altres funcions que requereixen l'ús de més components. En aquest apartat, es mostrarà un diagrama amb els que formen l'openplayer juntament amb la seva relació i en el següent punt es farà una explicació en major detall del funcionament de cadascuna de les parts.

1.1. Diagrama de blocs

En el següent diagrama es mostra el conjunt de parts que formen l'openplayer. A més, es poden observar un seguit de connexions que indiquen quin tipus de relació hi ha entre ells: *unidireccional* o *bidireccional*. Les connexions amb només un sol sentit indiquen una relació *unidireccional*, és a dir, que només és dirigida per un dels dos components (ex. comandament de TV, on el microcontrolador és l'únic encarregat d'enviar una sèrie de pulsos). En canvi, les connexions amb doble sentit indiquen una relació *bidireccional*, és a dir, existeix una interactivitat entre els dos components (ex. descodificador d'àudio, on aquest pot demanar al microcontrolador les següents dades d'una cançó o bé el microcontrolador pot donar l'ordre de canviar el nivell del volum).



Il.lustració 1.1: Diagrama de blocs amb la relació entre els diferents components que formen l'openplayer

2. Els components en detall

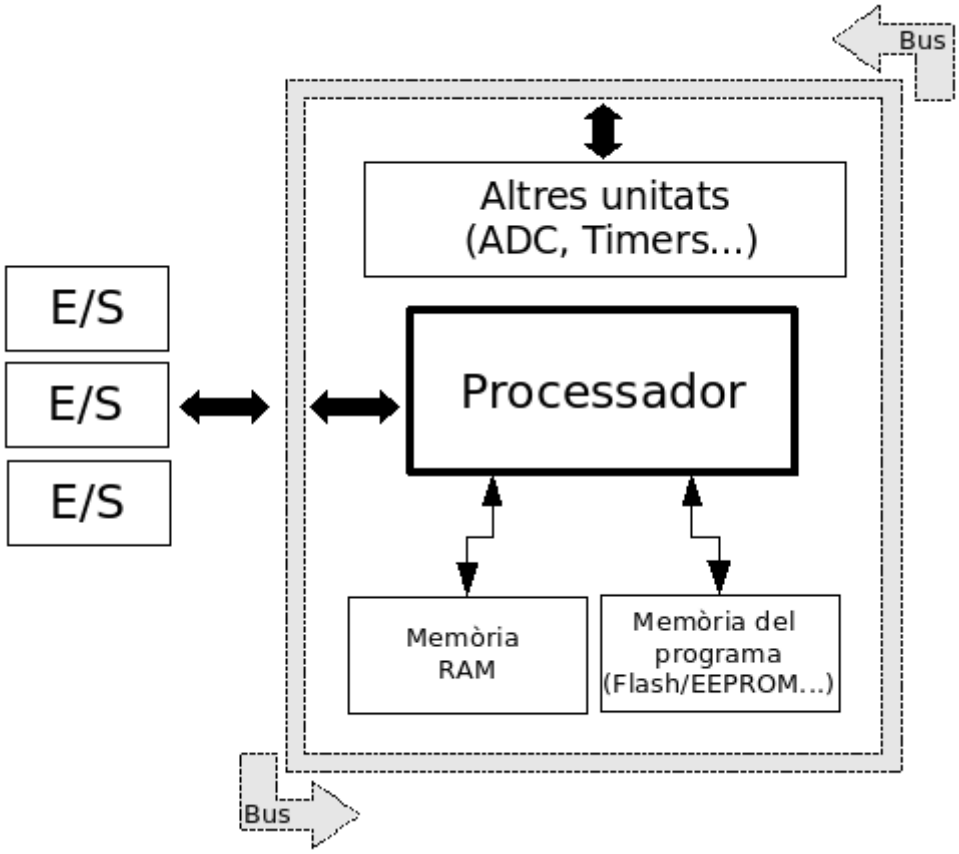
2.1. Microcontrolador

El microcontrolador és la part essencial de l'openplayer. Es podria qualificar com el seu "cervell". El microcontrolador és l'encarregat d'executar el firmware de l'openplayer i per tant és qui duu a terme la

majoria de tasques o bé dona ordres a components secundaris perquè en realitzin d'altres. En els següents punts es veurà la seva estructura bàsica i també el model utilitzat per l'openplayer.

2.1.1. Estructura d'un microcontrolador

Els microcontroladors són petits dispositius electrònics que ofereixen les tres parts essencials d'un ordinador integrades en un sol xip: processador, memòria i unitat d'entrades/sortides. A més, poden incloure altres utilitats com conversors analògic-digital (*ADC*), temporitzadors (*Timers*), entre moltes altres opcions que variaran en funció del camp d'aplicació del microcontrolador, el qual pot ser molt variat. Molt probablement estiguis envoltat de microcontroladors ja que són utilitzats en qualsevol aparell digital: rentadores, telèfons, microones... La seva arquitectura és més simple que la d'un ordinador convencional, tot i que això no significa en cap cas que no puguin realitzar tasques complexes. A la Il·lustració 2.1 es pot observar l'esquema bàsic d'un microcontrolador. El processador és el nucli del sistema i és l'encarregat de processar les instruccions del programa que ha carregat posteriorment l'usuari. Aquest programa es desa en una memòria no volàtil (Flash, EEPROM...) i utilitza la memòria RAM per desar variables (nombres, cadenes de text...) mentre es troba en execució. El programa també pot posar en funcionament altres unitats com els *ADC* i *Timers* o llegir els ports d'Entrada/Sortida (*E/S*).



Il.lustració 2.1: Esquema bàsic d'un microcontrolador

2.1.2. El microcontrolador utilitzat per l'openplayer

Il.lustració 2.2: Atmega1281

Com he mencionat anteriorment existeix una gran varietat de microcontroladors. Per l'openplayer vaig escollir una sèrie d'ús general anomenada *AVR* fabricada per l'empresa *Atmel* perquè són relativament simples de programar i les característiques que ofereixen són molt interessants. La majoria de reproductors de música no utilitzen aquest tipus de microcontroladors. Algunes empreses (*Atmel* per exemple) fabriquen xips especialment per a reproductors multimèdia que inclouen en el seu interior tots els components essencials (descodificador, controlador USB...) per tal de reduir costs i també espai en el circuit. El problema és que són poc documentats pel públic i per obtenir suport o exemples de codi és

necessari fer-ho a través de l'empresa que els produeix, cosa que pot suposar un cost afegit i en el meu cas el pressupost era baix.

L'openplayer funciona amb el model *Atmega1281* de la sèrie *AVR* tot i que també està preparat per funcionar amb l'*Atmega2561*. A la Taula 1 es mostren les característiques que ofereix aquest microcontrolador.

Característica	Descripció
Fabricant	Atmel
Model	Atmega1281
Arquitectura	RISC, 8-bits
Velocitat	1MIPS per cada Mhz. Fins a 16MIPS (a 16MHz - 5V). 8MIPS en el cas de l'openplayer, ja que opera a 3.3V i el rellotge màxim és de 8Mhz per aquest voltatge.
Memòries	- SRAM: 8KB - Flash: 128KB - EEPROM: 4KB
E/S	54 dels 64 pins disponibles poden funcionar com a E/S d'ús general
Utilitats incloses	- Timers de 8 i 16 bits - Canals de PWM (*Pulse With Modulation*) (8 i 16 bits) - ADC (8 canals), resolució màxima de 10 bits - Controlador SPI (*Serial Peripheral Interface*) - Controlador USART (*Universal Synchronous and Asynchronous Receiver and Transmitter*) - Interrupts externs - Modes de treball de baix consum
Porgramació i depuració	- ISP (*In System Programming*) permet programar el firmware del xip sense necessitat de desconnectar-lo del circuit. - JTAG (*Join Test Action Group*): Facilitat per programar el xip i depurar en temps real, entre d'altres funcions.

Taula 1: Característiques del microcontrolador utilitzat per l'openplayer

Com es pot observar, la sèrie AVR té uns límits importants, el principal és la velocitat. A l'openplayer ha estat un factor clau, ja que en algunes aplicacions es treballa al límit. A més, això també impossibilita crear un codi clarament estructurat i amb abstraccions ja que suposaria executar un nombre força més gran d'instruccions. Tot i això penso que el resultat ha estat satisfactori.

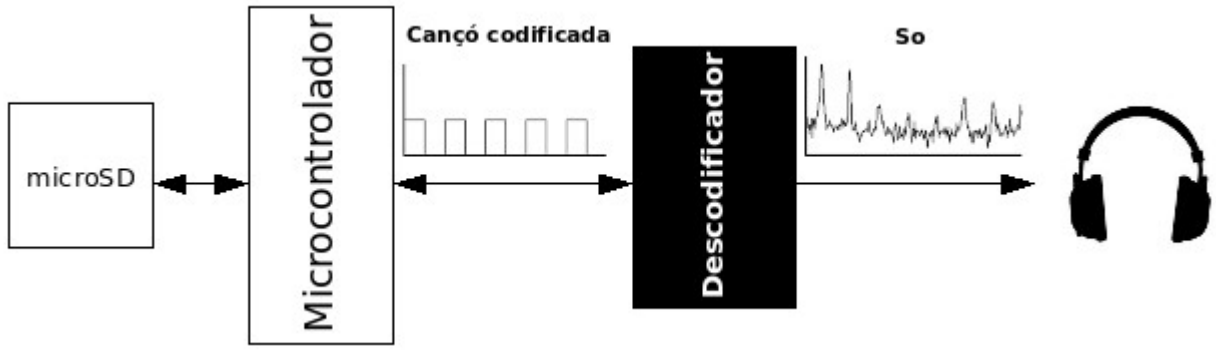
2.2. Descodificador d'àudio

Possiblement us haureu preguntat com pot ser que l'openplayer amb un microcontrolador de tan poca potència reprodueixi música d'un format comprimit que per tant, s'ha de descomprimir per poder ser escoltat. La resposta sorgeix gràcies a un altre component que realitza aquesta tasca: el descodificador d'àudio. Sense ell seria impossible crear un reproductor de música comprimida amb un microcontrolador *AVR*.

2.2.1. Funcionament del descodificador

La majoria de dispositius portàtils que reprodueixen algun format d'àudio o vídeo comprimits ho fan a través d'un altre component extern. Això és el que es coneix en el món de la informàtica com a *processament per hardware*. En un ordinador comú s'hi pot trobar un cas molt clar: la targeta gràfica.

Aquesta és l'encarregada de processar tasques relacionades amb els gràfics rebudes per part del processador i després mostrar-les en una pantalla. Això estalvia un gran treball al processador i per tant pot dedicar els seus recursos a altres coses. En el cas de l'openplayer es tracta de la mateixa idea, però fent-ho amb els fitxers de música. A la Il·lustració 2.3 es pot observar de manera esquemàtica aquest procés. Les cançons prèviament desades a la targeta microSD, són llegides pel microcontrolador i després enviades al descodificador perquè les descodifiqui i les transformi en so que l'usuari podrà escoltar mitjançant uns auriculars.



Il.lustració 2.3: Procés de descodificació d'un fitxer de música

2.2.2. El descodificador utilitzat per l'openplayer

Escollir un descodificador va ser relativament fàcil ja que les opcions eren ben poques. Alguns requerien la utilització d'un tercer xip (*DAC*, conversor digital-analògic) que transformava la cançó descodificada en so analògic. En canvi, una altra empresa anomenada *VLSI Solution Oy* especialitzada en aquest camp, disposava d'un conjunt de descodificadors que portaven aquest tercer xip integrat en el propi descodificador que a més, oferia altres possibilitats molt interessants.

Il.lustració 2.4: VS1053

La versió de sobretaula de l'openplayer, utilitza una versió del xip diferent que la de l'openplayer final[1]. El seu funcionament és essencialment el mateix, excepte alguns detalls. La Taula 2 mostra un resum de les característiques incloses pel descodificador de l'openplayer final.

Característica	Descripció
Fabricant	VLSI Solution Oy
Model	VS1053
Formats suportats	MP3, OGG, WMA, AAC, WAV, MIDI
Funcions	- Ajustatge del volum per a cada canal (D/E) - Ajustatge de Baixos/Aguts - Possibilitat de carregar programes a la seva memòria - Possibilitat de codificar veu entrada des d'un micròfon

Taula 2: Característiques del descodificador utilitzat per l'openplayer

2.3. Targeta microSD

La targeta microSD (família de les targetes *SecureDigital* però de tamany més petit) és el lloc on es desen les cançons, imatges i la resta de continguts que posteriorment són llegits per l'openplayer. De fet, s'hauria de considerar un component extern ja que no forma part del circuit: és tan sols una targeta extraïble. L'usuari per tant decideix quina és la capacitat de l'openplayer depenent de la targeta que hi

1 Vegeu Capítol 3, apartat 2.3

posa.

2.3.1. Capacitats suportades

Quan es van dissenyar per primer cop les especificacions que regeixen el funcionament de les targetes SD, només es va preveure una capacitat màxima de 2GB. Actualment però, aquest límit ha quedat petit hi ha calgut una actualització d'aquestes especificacions. Les targetes superiors a 2GB i amb un límit teòric de 32GB passen a anomenar-se SDHC (*SD High Capacity*) i pateixen uns petits canvis pel que fa a la seqüència d'inicialització i el mode d'accés a la memòria. Fins fa poc l'openplayer no les suportava, però el codi actual ha estat actualitzat i provat perquè en pugui fer ús. Per motius tècnics[2], aquesta actualització també obliga a canviar el sistema de fitxers a FAT32 (en les anteriors s'utilitzava FAT16) tot i que no suposa cap problema ja que el controlador de l'openplayer és dual (FAT16/32).

Il.lustració 2.5: microSDHC

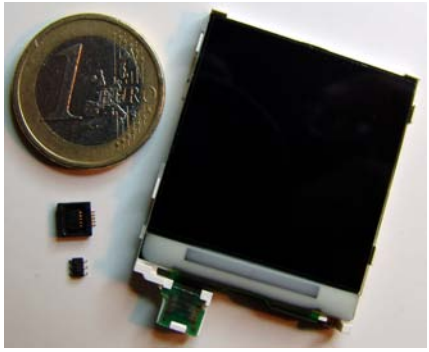
2.3.2. L'interior duna targeta SD

Existeixen molts models de memòries semblants a la SD: *MemoryStick, MMC, xD, CompactFlash...* Cal distingir-ne però dos tipus: les que incorporen controlador i les que no. En una memòria on s'incorpora controlador no es desa la informació directament a les cel·les de memòria, sinó que aquest exerceix com a intermediari i per tant és qui realment escriu el contingut a la cel·la corresponent. Les targetes SD inclouen controlador, cosa que en fa una gestió molt més simple. A la Il·lustració 2.6 es pot observar un esquema junt amb una fotografia real del cas de les targetes microSD i SD, on tan sols canvia el tamany i la forma ja que internament són iguals.

Il.lustració 2.6: Interior d'una targeta microSD i SD respectivament

2.4. Pantalla

Escollir una pantalla per l'openplayer no va ser fàcil. Necessitava un model simple que s'adaptés bé a la velocitat del microcontrolador però alhora de bona qualitat i que disposés d'una documentació completa. Navegant vaig trobar-me amb un projecte anomenat *Yampp*, un altre reproductor que utilitzava entre diverses pantalles una que era en color. Vaig informar-me sobre aquell model i resultava ser la pantalla utilitzada en alguns models del mòbil *Nokia 6610*. Era una pantalla simple, que incloïa un microcontrolador intern i que estava totalment documentada, així que vaig decidir provar sort amb aquell model.



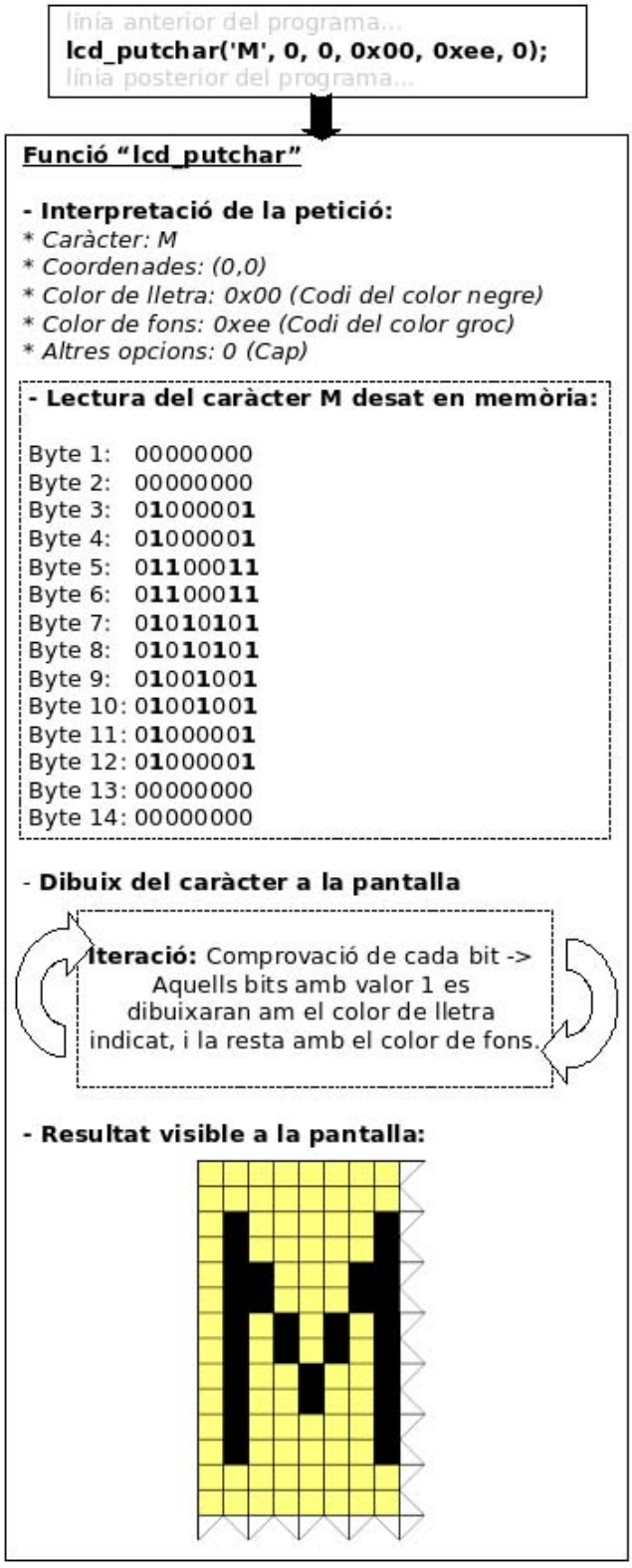
Il.lustració 2.7: Pantalla

2.4.1. Funcionament i característiques

La pantalla utilitza un microcontrolador intern fabricat per l'empresa *LEADIS*, el qual ofereix una

2 Vegeu Annex 5, "Altres documents: Sistemes de fitxers FAT16/FAT32"

gestió molt simple de la pantalla. Mitjançant una sèrie de comandes es dibuixen els píxels de la pantalla, s'ajusta el contrast, el nombre de colors en què es vol treballar, etc. En cap cas la pantalla inclou funcions incorporades per mostrar text o dibuixar formes geomètriques. Aquestes són dibuixades píxel a píxel pel firmware de l'openplayer. A la Il·lustració 2.8 es pot observar un exemple de quin es el procés que es segueix quan el programa demana mostrar la lletra "M" a la coordenada (0,0) de la pantalla.



Il.lustració 2.8: Procés dut a terme per dibuixar la lletra "M" a la pantalla

La il·lustració anterior és un bon exemple per comprovar que quan es veu un menú (icones, texts dels

diferents apartats, requadres...) posteriorment s'ha dut a terme un llarg procés per dibuixar tot aquest contingut. És clar però que per l'ull humà és impossible percebre aquests processos, ja que es duen a terme a una gran velocitat.

La pantalla té certes limitacions, com per exemple el fet que la comunicació sigui via sèrie. Això sumat amb la poca velocitat del microcontrolador principal pot suposar un problema en moments on cal un gran nombre d'actualitzacions del contingut en poc temps. És per aquest motiu que la pantalla només treballa a 256 colors (1 byte per píxel) i no als 4096 (2 bytes per píxel) que ofereix per tal d'anar més ràpid. Tot i això, és totalment suficient per l'openplayer. Algunes de les característiques que ofereix aquesta pantalla es mostren a la Taula 3.

Característica	Descripció
Controlador intern	LDS176
Mètode de comunicació	Bus sèrie
Resolució	132x132 píxels
Nombre de colors	Màxim de 4096. L'openplayer treballa en el mode de 256 colors.
Il·luminació	Sí, però controlada externament
Altres	- Contrast ajustable digitalment - Possibilitat d'apagar la pantalla però conservant el contingut per reduir el consum

Taula 3: Característiques de la pantalla utilitzada per l'openplayer

2.4.2. Regulació de la intensitat de llum

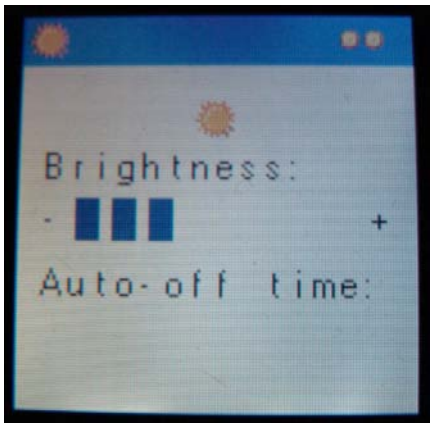
La pantalla que vaig seleccionar inclou 4 LEDs blancs que il·luminen el seu fons, però aquests necessiten funcionar com a mínim a 6V. Això pot semblar un problema, ja que el circuit treballa només a 3.3V i si es connecten a aquesta tensió no s'encendran. Per aquest motiu cal fer ús d'un petit xip extern que augmenti el voltatge del circuit, però que a més, ofereixi la possibilitat de regular la intensitat de la llum. En el cas de l'openplayer s'ha seleccionat el xip *LT1932* produït per *Linear Technology* (visible a la Il·lustració 2.7). El fet que sigui regulable també tindrà conseqüències importants en el consum de bateria. A la Taula 4 es pot veure el consum en relació a la intensitat de llum mesurat mentre l'openplayer es trobava mostrant el menú principal.

Intensitat de llum	Consum
20%	43mA
60%	53mA
100%	65mA

Taula 4: Consum en relació a la intensitat de llum

Tal com es pot comprovar doncs, és molt important tenir en compte en quina intensitat es treballa, ja que això ens podria ajudar a allargar la durada de la bateria.

L'openplayer també ofereix una utilitat gràfica on ajustar aquesta intensitat. A més, quan s'ajusta el nou valor és desat i restablert quan es torna a engegar el reproductor. Una altra funció interessant és l'auto-apagat: després d'un període d'inactivitat (ex. l'usuari no pulsa cap tecla) la llum s'apaga automàticament per reduir el consum.



Il.lustració 2.9: Utilitat per ajustar la intensitat de llum

2.5. Bateria

Perquè un dispositiu pugui ser portàtil i per tant autònom es requereix l'ús d'una bateria. Actualment existeixen bateries de llarga durada i d'un tamany molt reduït. En el cas de l'openplayer vaig escollir bateries de Li-Ion ja que compleixen aquestes dues característiques. Tot i així no n'hi ha prou de connectar la bateria al circuit: es necessita regular el seu voltatge perquè s'ajusti al valor que es treballa i també incloure un carregador, que no es tracta d'un simple endoll com es veurà a continuació.

2.5.1. Bateries de Li-Ion

Les bateries de Li-Ion (*Lithium-Ion*) són les més utilitzades pels aparells electrònics en l'actualitat. Se'n poden trobar a qualsevol ordinador portàtil, càmera digital, telèfon mòbil... La utilització d'aquestes bateries ofereix un gran nombre d'avantatges:

- Elevada densitat d'energia emmagatzemada en el mateix espai respecte altres tipus de bateries
- Poc pesants i de tamany reduït
- El voltatge nominal ofert per cada cel·la de la bateria és de 3.7V
- La tassa d'àutodescàrrega és baixa (només d'un 6% cada mes)

Cal destacar també alguns dels seus inconvenients:

- Vida curta: Un màxim de 3 anys o bé entre 300 i 600 cicles de càrrega/descàrrega[3]
- Possibilitat de sobreescalfaments (fins i tot explosions) si no són degudament controlades

Les bateries de Li-Ion acostumen a indicar en mAh (unitat de càrrega) la seva capacitat. Gràcies a aquest valor es pot determinar la durada de la bateria si se sap el consum del dispositiu electrònic mitjançant la fórmula següent:

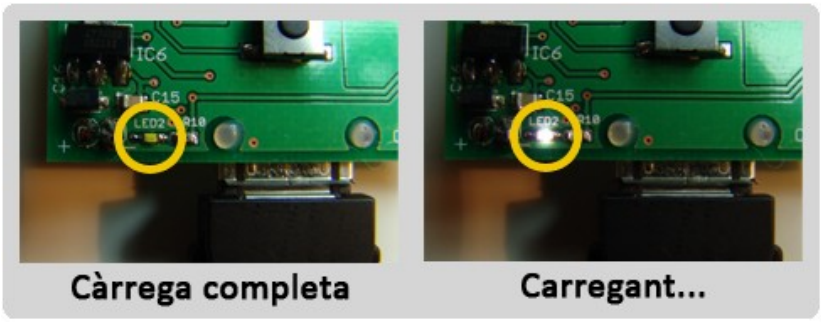
$$Durada\,de\,la\,bateria(h) = \frac{[Capacitat\,de\,la\,bateria(mAh)]}{[Consum\,del\,dispositiu(mA)]}$$

2.5.2. Càrrega de la bateria

El procés de càrrega d'una bateria de Li-Ion és un procés simple però delicat, ja que ha de ser molt precís. La tensió final de càrrega pot ser de 4.1V o 4.2V, depenent de la composició de l'ànode de la bateria (grafit o carbó respectivament). Durant els primers minuts, el voltatge de la bateria augmenta molt

3 Aquests valors poden variar molt ja que la bateria es pot veure deteriorada per altres factors com la temperatura de treball, una intensitat molt elevada i contínua durant la seva utilització...

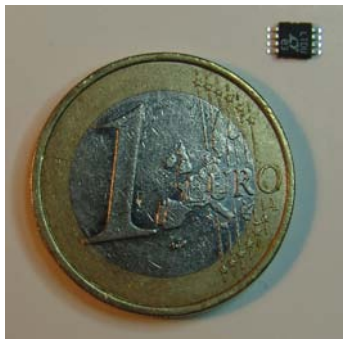
ràpidament fins arribar gairebé al voltatge de càrrega total. La diferència entre aquest i el final pot ser de tant sols 0.1V, cosa que fa que si no es disposa d'un carregador precís, es podria considerar que la bateria està carregada quan en realitat es troba molt per sota de la càrrega total. Així doncs és convenient utilitzar xips especialitzats en la càrrega de bateries de Li-Ion. En el cas de l'openplayer, s'ha seleccionat el *MAX1811*, un xip fabricat per *Maxim* que permet la càrrega de bateries d'una sola cel·la i que a més permet seleccionar el voltatge de càrrega (4.1V o 4.2V), la intensitat de càrrega (100mA o 500mA) i també col·locar en el circuit un LED indicador de l'estat de càrrega (Il·lustració 2.10).



Il.lustració 2.10: LED indicador de l'estat de càrrega

2.5.3. Regulació del voltatge del circuit

Ja que el circuit necessita una tensió constant de 3.3V, és necessari utilitzar un regulador de voltatge que connectat a l'entrada de la bateria ens ofereixi aquesta tensió de manera estable (sempre i quan la tensió de la bateria sigui igual o superior a 3.3V). En el cas de l'openplayer, s'utilitza el model *LT1521-3.3* de *Linear Technology*.



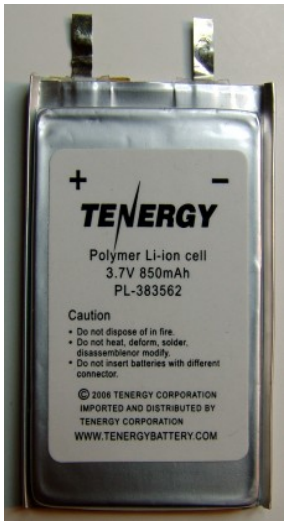
Il.lustració 2.11: LT1512-3.3

2.5.4. Bateria seleccionada per l'openplayer

La bateria que inclou l'openplayer ofereix les característiques que es mostren a la Taula 5. La seva selecció va venir donada en funció a la seva capacitat, pes i tamany com també pel seu preu.

Característica	Descripció
Fabricant	TENERGY
Capacitat	850mAh
Pes	20g
Voltatge de càrrega	4.2V
Voltatge totalment descarregada	2.75V
Corrent màxim de consum	2600mA

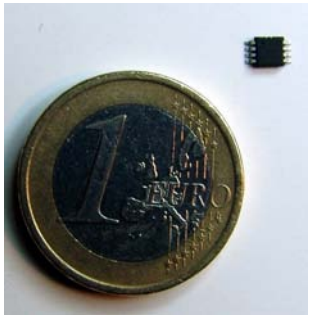
Taula 5: Característiques de la bateria utilitzada per l'openplayer



Il.lustració 2.12: Bateria utilitzada per l'openplayer

2.6. Sensor de temperatura

Quan vaig començar amb el projecte de l'openplayer, mai hauria pensat que aquest inclouria un sensor de temperatura. Va ser gràcies a *Maxim,* que per error va enviar-me unes mostres equivocades i justament eren les d'un sensor de temperatura digital, concretament el model *DS18B20*. Així doncs vaig decidir aprofitar l'oportunitat i incloure'l a l'openplayer. La Taula 6 mostra algunes de les característiques d'aquest sensor.

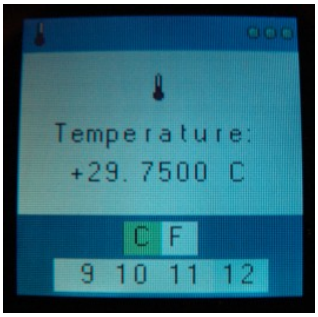


Il.lustració 2.13: DS18B20

Característica	Descripció
Fabricant	Maxim
Model	DS18B20
Rang de temperatures	-55ºC a +125ºC
Resolució	Seleccionable: 9,10,11 i 12 bits
Precisió	±0.5ºC

Taula 6: Característiques del sensor de temperatura utilitzat per l'openplayer

L'openplayer incorpora una petita aplicació que mostra a la pantalla la temperatura llegida pel sensor on també es pot seleccionar la resolució i l'escala (ºC o ºF).



Il.lustració 2.14: El termòmetre inclòs per l'openplayer

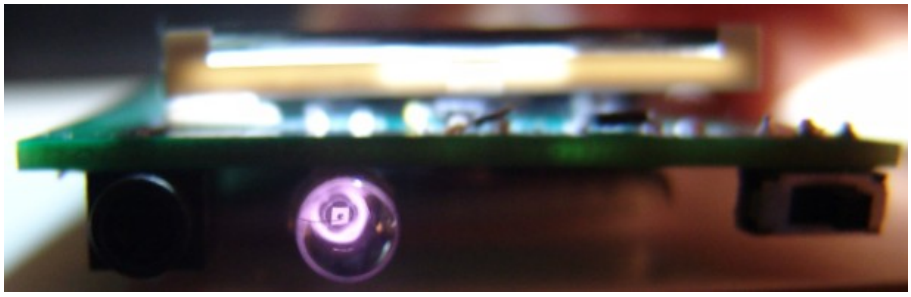
2.7. Comandament de TV

Aquesta era una funció que tenia prevista incloure per un motiu molt simple: la televisió de la meva habitació té el comandament espatllat. És veritat que venen comandaments universals, però ja que estava fent un reproductor de música vaig pensar que afegir-hi un comandament de televisió ajudaria a completar-lo i de passada podria aprendre com funcionen.

2.7.1. Funcionament dels comandaments de TV

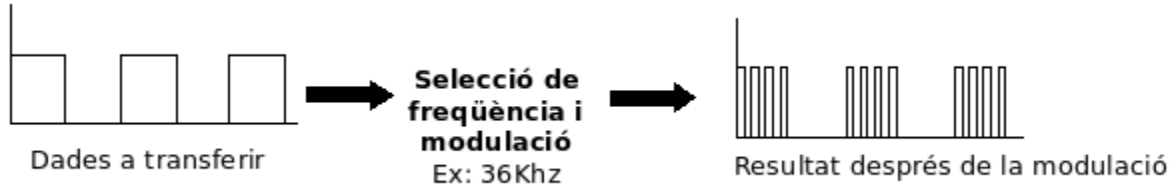
El funcionament dels comandaments de televisió es basa en la transferència d'una sèrie de pulsos mitjançant llum infraroja que després són rebuts i interpretats per un receptor col·locat a la televisió. Si la petició enviada és interpretada com a vàlida llavors la televisió executarà l'acció corresponent.

La llum infraroja és una llum amb una longitud d'ona entre 400nm i 700nm, fora de l'espectre de llum visible per l'ull humà. És per aquest motiu que quan es pulsa alguna tecla del comandament no s'observa cap fenomen, quan en realitat s'estan transmetent una sèrie de pulsos de llum a gran velocitat. Es pot comprovar aquest fet enfocant l'objectiu de qualsevol càmera digital cap a una font de llum infraroja com és el comandament d'una televisió (Il·lustració 2.15).



Il.lustració 2.15: Llum infraroja del comandament de l'openplayer captada mitjançant una càmera digital

Tot i això existeix un altre un problema: en el nostre entorn sovint es poden trobar fonts de llum infraroja que podrien causar interferències a les peticions enviades pel comandament (ex. sol, cossos calents...) i per tant es necessita modular la petició a una freqüència determinada (Il·lustració 2.16). D'aquesta manera, el receptor podrà descartar qualsevol altra font indesitjable. Cal destacar també que el receptor torna a fer el procés invers, és a dir, desmodula la senyal un cop rebuda perquè pugui ser utilitzada altre cop.



Il.lustració 2.16: Procés de modulació

2.7.2. El protocol de comunicació

Els pulsos transmesos per un comandament no son aleatoris, sinó que segueixen un protocol. Actualment existeixen molts protocols, ja que moltes marques han decidit crear-ne un de propi pels seus dispositius. Tot i això, un dels protocols més estesos i utilitzat per diverses marques és el conegut com a RC-5, creat per Phillips. Aquesta és la raó que em va portar a incloure'l a l'openplayer, tot i que està preparat per afegir-ne d'altres en un futur si es desitja. La Taula 7 mostra un resum de les característiques d'aquest protocol.

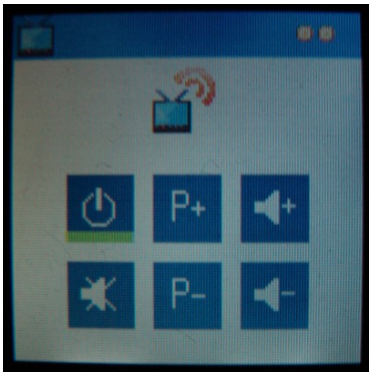
Característica	Descripció
Freqüència de modulació	36Khz
Nombre de peticions disponibles	64
Nombre d'aparells que es poden controlar	32 *- Una mateix codi de petició petició pot dirigir-se a un tipus d'aparell determinat (ex: TV, vídeo, minicadena...).*

Taula 7: Característiques del protocol RC-5

El codi de cada petició com també el codi dels aparells són valors definits per les especificacions del protocol RC-5.

2.7.3. El comandament inclòs a l'openplayer

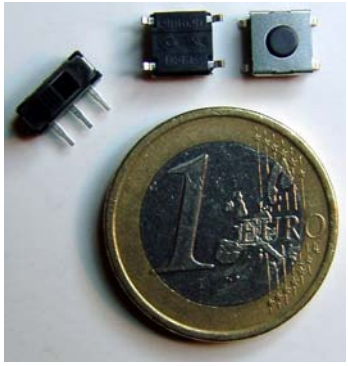
L'openplayer inclou una aplicació gràfica que permet executar les 6 comandes bàsiques d'un comandament de televisió: apagar/engegar, anul·lar volum, canviar de canal (+ i -) i ajustar el volum (+ i -). Tal com indicava abans, el codi està preparat per afegir nous protocols utilitzant la mateixa aplicació. Possiblement en noves versions del firmware s'incloguin més protocols (Sony, Sharp...).



Il.lustració 2.17: L'alicació "TV Remote" inclosa a l'openplayer

2.8. Teclat

El teclat de l'openplayer no només està format per les 5 tecles principals, sinó que també són considerades tecles el botó de reset i la tecla de bloqueig. Aquests últims potser són menys utilitzats però són del tot necessaris.



Il.lustració 2.18: Tecles de l'openplayer

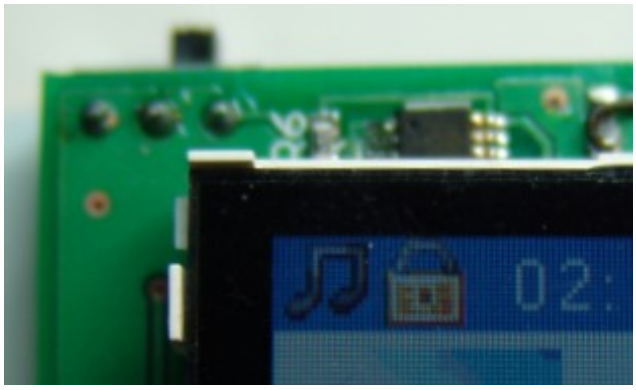
2.8.1. El teclat principal

El teclat principal consta de 5 tecles localitzades sota la pantalla que ofereixen el control del reproductor. Cada aplicació les utilitza segons el seu criteri però de la manera més lògica possible perquè l'usuari pugui conèixer les seves funcions sense cap manual d'instruccions.

2.8.2. La tecla de bloqueig

La tecla de bloqueig pot adquirir dues funcions:

1. **Bloqueig**: Mentre el reproductor es troba en funcionament es pot canviar a la posició de la dreta perquè el reproductor quedi bloquejat (Il·lustració 2.19). D'aquesta manera, les 5 tecles principals queden inutilitzades i apareix el símbol d'un candau a la part superior de la pantalla. És una funció útil en el moment que es porta el reproductor a la butxaca i no es vol que les tecles es pulsin de manera indesitjada.



Il.lustració 2.19: Openplayer amb la tecla de bloqueig activa

2. **Parada/Engegada**: Per tal de parar l'openplayer (Menu principal > Sortir > Apagar) es demanarà que es bloquegi, per tant la tecla de bloqueig quedarà a la posició de la dreta. En el moment que es desitgi tornar a engegar el reproductor, només caldrà moure la tecla altre cop a la posició de l'esquerra.

2.8.3. Els botons de reset

A la part del darrera de l'openplayer es poden observar dues tecles petites. Una serveix per reiniciar el el microcontrolador en cas que es pengi o presenti alguna anomalia (ex. no s'engega). L'altre també fa la mateixa funció, però fa el reset per tots els components del circuit. Per fer-ho deixa tot el circuit sense corrent, cosa que provoca la parada de tots els components mentre es troba pulsat i en deixar-se anar s'engeguen tots altre cop al tornar a tenir corrent.

2.9. Dock

L'openplayer havia de contenir algun connector que a través d'ell em permetés programar-lo, carregar la bateria i que a més oferís extensibilitat. Va ser complicat trobar un connector petit i que a més fos soldable a ma (tan la base del connector com el connector amb cable). Finalment vaig trobar model que tenia aquestes característiques (Il·lustració 2.20) fabricat per l'empresa *Hirose Electric*. El model oferia el connector amb més o menys pins. Vaig optar pel connector de 10 pins, però com que en aquell moment es trobava fora d'estoc, vaig haver de comprar el model de 18 pins. Aquest connector ha permès a l'openplayer oferir les funcions que es descriuen en els següents punts.

Il.lustració 2.20: ST60-18P

2.9.1. Programació i depuració

Tal com es descriu a l'apartat 2.1.2, el microcontrolador que utilitza l'openplayer ofereix la possibilitat de ser programat sense haver de ser desconnectat del circuit. A través del Dock i amb el cable de programació corresponent, es podrà actualitzar o carregar de nou el firmware de l'openplayer.

També es poden utilitzar les línies del bus USART connectades a l'ordinador com a font de depuració utilitzant les funcions incloses en el mòdul <*debug/usart.h*> del codi de l'openplayer.

2.9.2. Càrrega de la bateria

Connectant el cable de càrrega al Dock es podrà carregar la bateria de l'openplayer. L'entrada de la bateria ha de provenir d'una font d'alimentació de corrent contínua com a mínim de 6.5V i amb un màxim de 15V. La intensitat mínima del carregador haurà de ser de 500mA, tot i que és possible canviar la intensitat a 100mA a través de la configuració de l'openplayer.

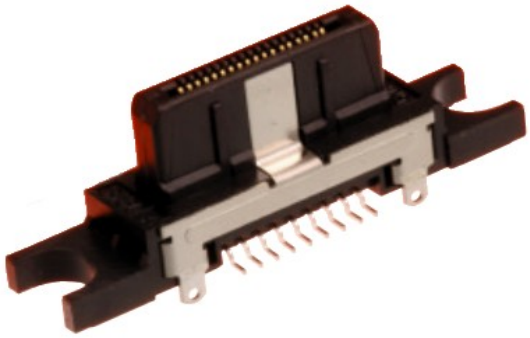
2.9.3. Extensibilitat i alimentació

En el Dock s'hi troben dues línies corresponents a les del bus USART del microcontrolador de l'openplayer (Rx i Tx) . Això permet crear dispositius externs que es puguin comunicar amb el microcontrolador i per tant oferir noves funcions. Per afegir-ne, només caldria carregar un nou firmware preparat pel dispositiu en qüestió. A més, aquest dispositiu extern pot utilitzar la bateria de l'openplayer com a font d'alimentació (3.3V) amb un límit de consum d'aproximadament 200mA. Alguns exemples de possibles extensions es mostren a continuació:

- Sensors múltiples (pressió, brúixoles...)
- Llanterna de baixa potència
- Receptor infraroig per controlar remotament el reproductor

2.9.4. Sortides d'àudio

També s'inclouen sortides d'àudio en el Dock. Això permetria crear uns altaveus amb el connector vertical de la base corresponent a la de l'openplayer (Il·lustració 2.21), on es podria escoltar la música que s'està reproduint sense utilitzar cap cable. És un producte comú en el cas dels iPod fabricats per Apple (Il·lustració 2.22).

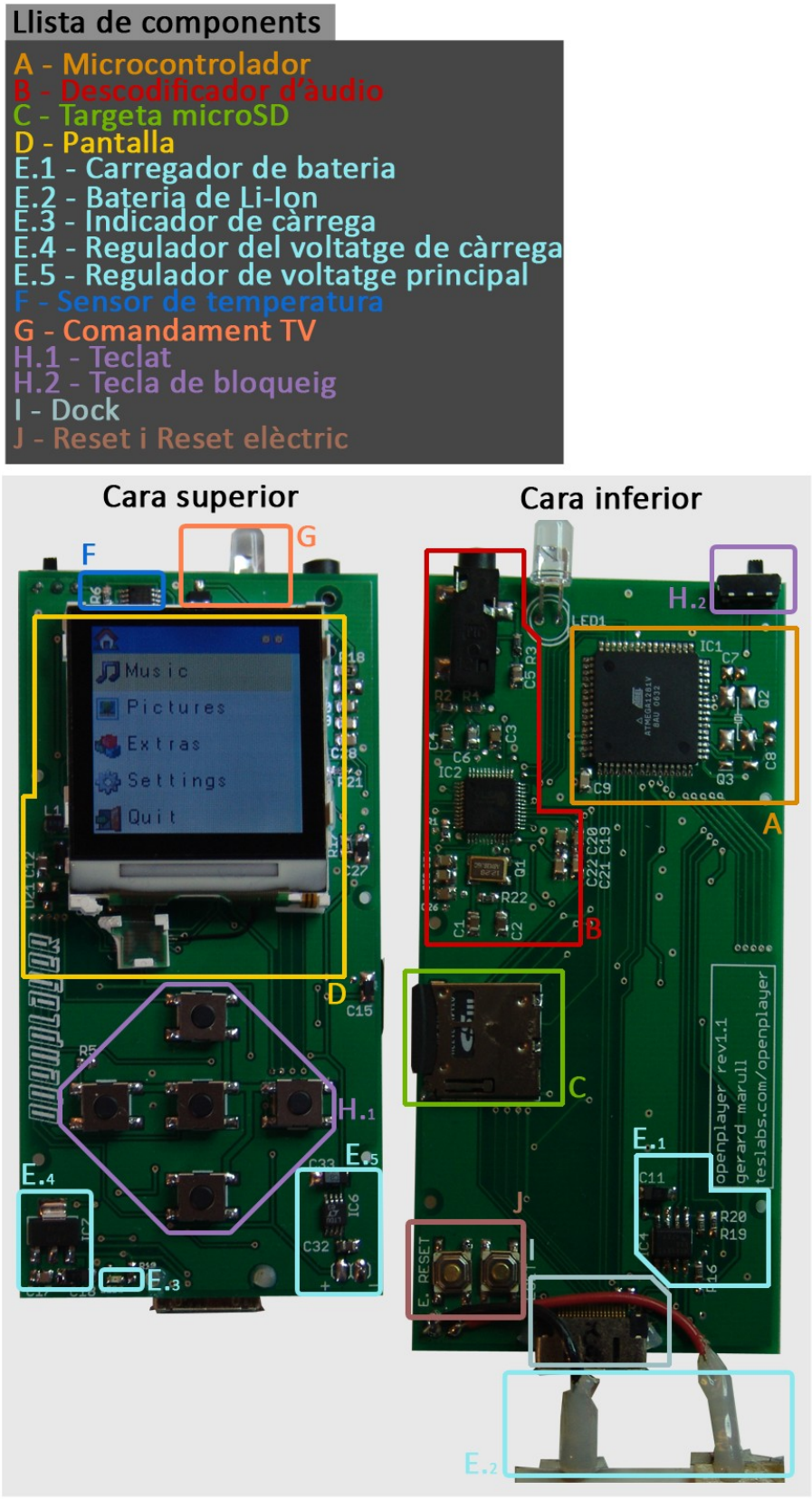


Il.lustració 2.21: ST80X-18S, connector vertical pel dock de l'openplayer

Il.lustració 2.22: Exemple d'uns altaveus amb base

3. Identificació en el model real

Penso que és important obtenir una idea gràfica de com es troben disposats tot el conjunt de components que s'acaben de descriure en el reproductor real. En la següent fotografia s'obté una concepció força més realista que no pas únicament utilitzant esquemes.



Capítol 2

Programació

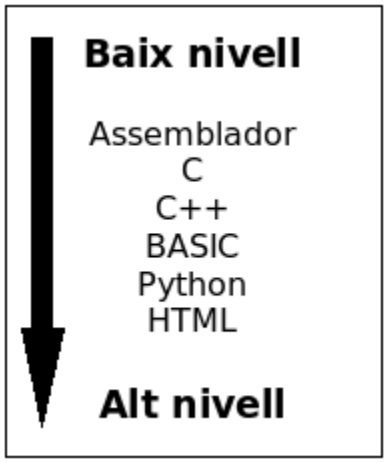
1. Introducció

Anteriorment havia mencionat en algun cas la paraula firmware. El firmware, no és res mes que el programa que fa funcionar l'openplayer. Aquest està compost pel conjunt d'instruccions compatibles amb el microcontrolador de l'openplayer que són executades seguint un ordre lògic un cop es troben carregades al microcontrolador. Tot aquest conjunt d'instruccions però, ha hagut de passar per un procés previ de programació el qual s'explicarà en detall en els diferents punts d'aquest capítol.

1.1. Els llenguatges de programació

Cada arquitectura d'ordinador suporta una sèrie d'instruccions determinades i per tant si es canvia també s'haurà de reescriure qualsevol codi que s'hagi programat seguint el nou conjunt d'instruccions. Es pot programar directament mitjançant aquestes instruccions tot i que és un mètode poc corrent. Aquest "llenguatge" basat en les instruccions suportades per un processador s'anomena *assemblador*. Arquitectures com la d'alguns microcontroladors on el nombre d'instruccions disponibles és baix es poden trobar forces programes escrits en assemblador. En altres casos com els ordinadors actuals és un mètode molt poc freqüent, únicament utilitzat en llocs molt específics com per exemple el nucli d'un sistema operatiu. L'avantatge de l'assemblador és que es pot generar codi molt eficient, tot i que és un codi sense estructures i sense portabilitat, cosa que el fa molt incòmode per a projectes grans. Aquests inconvenients van portar a la creació dels llenguatges de programació que es coneixen avui dia, els quals són universals (no depenen d'una arquitectura) i per tant quan es crea un programa, portar-lo a altres arquitectures serà un procés molt més fàcil. És clar però que un ordinador mai podrà entendre un llenguatge d'aquest tipus i per tant, es necessitarà traduir-lo a les instruccions equivalents de l'arquitectura en qüestió. Aquest procés s'anomena *compilació*. També cal dir que alguns llenguatges en comptes de ser compilats són processats per un intèrpret en temps real (ex. HTML, Tcl/Tk...).

Els llenguatges de programació es classifiquen segons el seu *nivell d'abstracció*, és a dir, segons el seu nivell de relació amb el hardware. Quant més baix és el seu nivell, major és aquesta relació. L'assemblador és el llenguatge de més baix nivell possible, ja que treballa directament amb els registres del processador, adreces de memòria, etc. En altres llenguatges, com el C, s'ofereixen algunes característiques dels llenguatges de baix nivell (ex. punters de memòria) i per tant es podrien considerar intermedis. En canvi, altres llenguatges com el BASIC són d'alt nivell ja que no conserven cap característica de les anteriors.



Il.lustració 1.1: Classificació dels llenguatges segons el seu nivell

1.2. El llenguatge del codi de l'openplayer

1.2.1. Selecció d'un llenguatge

Quan se selecciona un llenguatge de programació es necessita sempre un compilador i una llibreria de programació (conjunt de funcions bàsiques) per la arquitectura on es pretén executar el nostre programa. En el cas del microcontrolador de l'openplayer les opcions que em vaig plantejar van ser les següents:

- Assemblador (sense necessitat de compilador ni tampoc llibreria)
- BASIC
- C

El primer, l'assemblador el vaig descartar ja que si la meva experiència en microcontroladors AVR era baixa, l'assemblador per AVR encara ho era més. BASIC era una opció però el compilador era propietari i el llenguatge no és massa eficient. La última opció va ser el llenguatge C, el qual disposa d'un compilador i llibreria lliures i amb el que havia treballat anteriorment cosa que les hores d'aprenentatge se simplificaven en només les característiques específiques pels microcontroladors AVR.

1.2.2. El llenguatge C

El llenguatge C va ser creat l'any 1969 per Ken Thompson i Dennis M. Ritchie als laboratoris Bell. El llenguatge C es considera de nivell mitjà, tot i que ofereix moltes característiques de baix nivell. És per aquest motiu que és possible generar compilacions molt eficients i per tant és utilitzant àmpliament en la programació de sistemes. De fet, C és considerat un dels llenguatges de programació més elegants que existeix. Un exemple famós és el nucli del sistema operatiu Linux, on gairebé tot el codi està programat en aquest llenguatge.

```
/* Exemple de codi en C per a microcontroladors AVR */

/* Inclusió de les capçaleres de la llibreria */
#include <avr/io.h>
#include <util/delay.h>

/* Funció principal */
int main(void){

          /* Configurem el el pin PA0 com a sortida */
          DDRA|=(1<<PA0);
          PORTA|=(1<<PA0);

          /* Alternem l'estat del pin PA0 per causar l'efecte d'intermitència*/
          while(1){
                    PORTA&=~(1<<PA0);
                    _delay_ms(65535);
                    PORTA|=(1<<PA0);
                    _delay_ms(65535);
          }

}
```

Codi 1: Exemple d'un programa programat en llenguatge C que fa apagar i encendre un LED indefinidament en un microcontrolador AVR.

2. Programari

Per poder programar el codi de l'openplayer ha estat necessària la utilització d'un conjunt d'eines de programació com el compilador, la llibreria de programació, el programador, l'entorn de desenvolupament, etc. En els següents punts s'explicarà en detall cadascun d'aquests programes. La plataforma de desenvolupament no afecta en absolut en el cas dels microcontroladors AVR, ja que tot el conjunt d'eines descrit es troba disponible per a les plataformes més comunes (Linux, MacOS X i

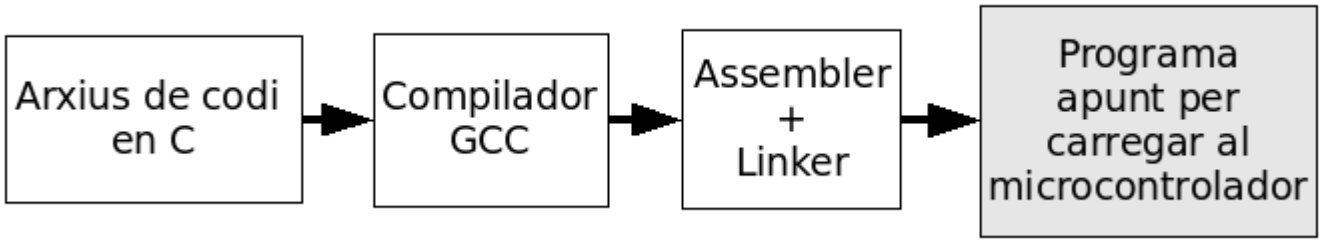
Windows). Tot el programari utilitzat durant el procés de creació de l'openplayer es pot trobar accedint a l'Annex 5, apartat 2.5.

2.1. Compilador i complements

El compilador per a llenguatge C utilitzat en el cas de l'openplayer és el GCC 4.2.1 (*GNU Compiler Collection*). És un compilador lliure molt estès i que genera compilacions molt eficients. De fet, el GCC el formen una col·lecció de compiladors de diversos llenguatges (C, C++, Java, Fortran...) i amb suport per diverses plataformes tot i que en aquest cas només s'utilitza la versió del C per microcontroladors AVR.

A més del compilador, qui és l'encarregat de fer la traducció del codi a assemblador, es necessiten uns altres programes: l'*assembler* i el *linker*. L'*assembler* és l'encarregat d'agafar la traducció i transformar-la a mode numèric (els famosos 0 i 1 dels ordinadors). El *linker*, és l'encarregat d'ajuntar els diversos fitxers que formen un codi.

Il.lustració 2.1: Logotip GCC



Il.lustració 2.2: Procés simplificat de compilació

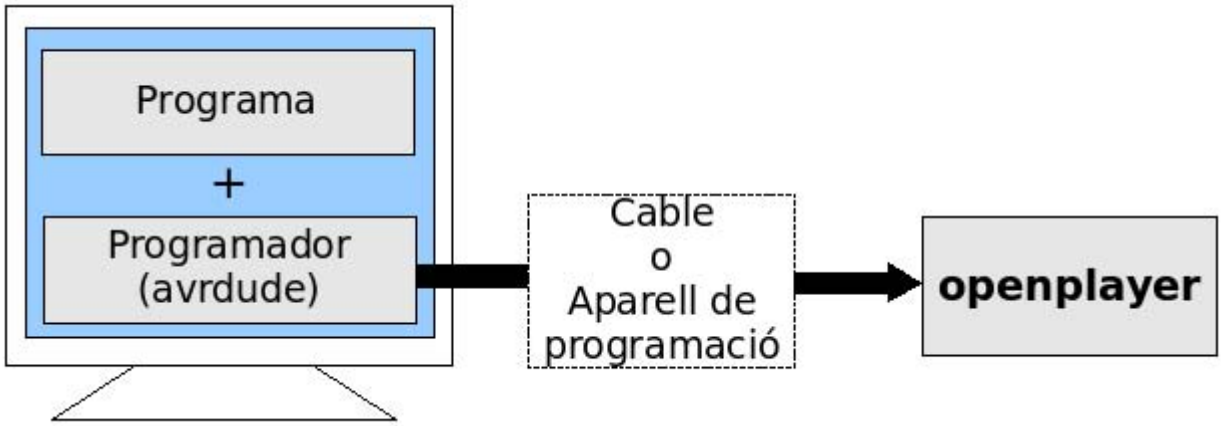
2.2. Llibreria de programació

En qualsevol entorn és necessari utilitzar una llibreria de programació. Aquesta proporciona les funcions bàsiques per a un programa, com per exemple les operacions matemàtiques complexes (sin, cos...), operacions d'E/S i algunes funcions específiques de l'entorn on es treballa. En el cas dels microcontroladors AVR i el llenguatge C existeix la llibreria *avr-libc*. Gràcies a ella es pot disposar de tot aquest conjunt de funcions, a més de capçaleres on s'especifiquen les adreces de memòria de tots els registres del microcontrolador, els codis d'inicialització, entre moltes altres opcions. Algunes de les funcionalitats que ofereix la llibreria avr-libc i que són utilitzades per l'openplayer són les següents:

- Definicions de les adreces i registres pel microcontrolador Atmega1281/2561 (<*avr/io.h*>)
- Facilitat d'accés a les memòries Flash i EEPROM (<*avr/pgmspace.h*> i <*avr/eeprom.h*>)
- Utilitats d'E/S (ex. família de funcions *printf*) (<*stdio.h*>)
- Utilitats per treballar amb cadenes de text (<*string.h*>)

2.3. Programador

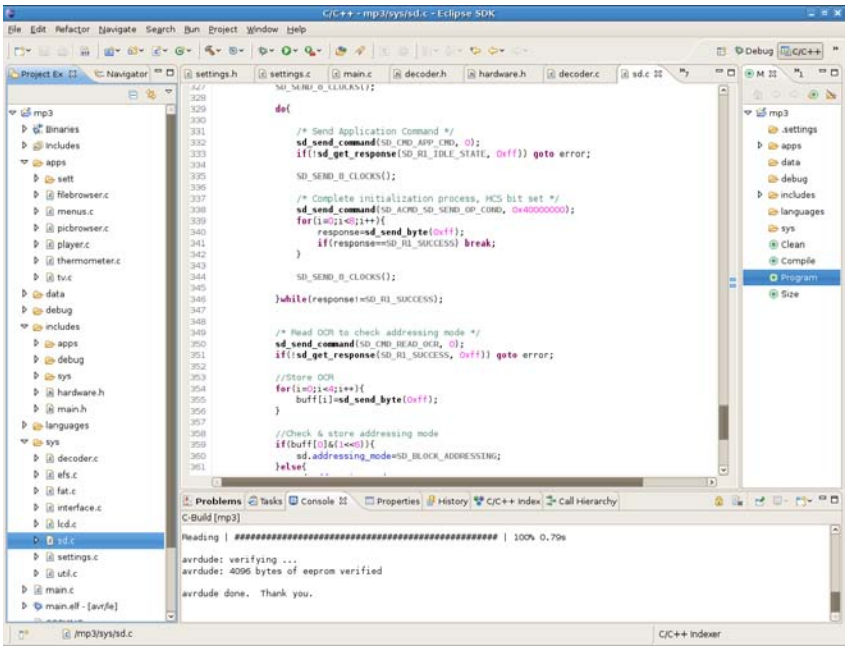
Un cop es disposa d'un compilador i llibreria de programació es poden crear programes i preparar-los per ser executats en el microcontrolador. Tot i així aquest programa el tenim a l'ordinador i no al microcontrolador. Mitjançant un cable o un aparell adequat per a la programació juntament amb un programa anomenat *programador* s'haurà de carregar el programa al microcontrolador. En el cas de l'openplayer s'utilitza l'*avrdud*e, el qual suporta gairebé la totalitat dels microcontroladors AVR actuals i també la majoria d'aparells de programació disponibles. L'*avrdude* a més, permet obtenir el contingut de la memòria actual del microcontrolador (Flash i EEPROM) i també ajustar alguns paràmetres del microcontrolador.



Il.lustració 2.3: Procés de càrrega del programa a l'openplayer

2.4. Entorn de desenvolupament

Per escriure i gestionar el codi, es necessita un programa que proporcioni un entorn òptim de desenvolupament. Sempre es pot utilitzar un simple editor de texts, però és un mètode molt incòmode. Els entorns de desenvolupament, també coneguts com a IDE (*Integrated Development Environment*) permeten en un mateix programa realitzar totes les tasques relacionades amb la programació: escriptura de codi, control de versions, compilació i càrrega del codi amb pocs clics, sintaxi del codi en colors per treballar més còmode, etc. En el cas de l'openplayer s'ha utilitzat l'*Eclipse* juntament amb el seu complement *CDT* per a programar en C o C++. En aquest cas l'usuari podria seleccionar altres entorns segons les seves necessitats o característiques de l'ordinador ja que no ha de ser cap programa específic.



Il.lustració 2.4: L'entorn de desenvolupament Eclipse

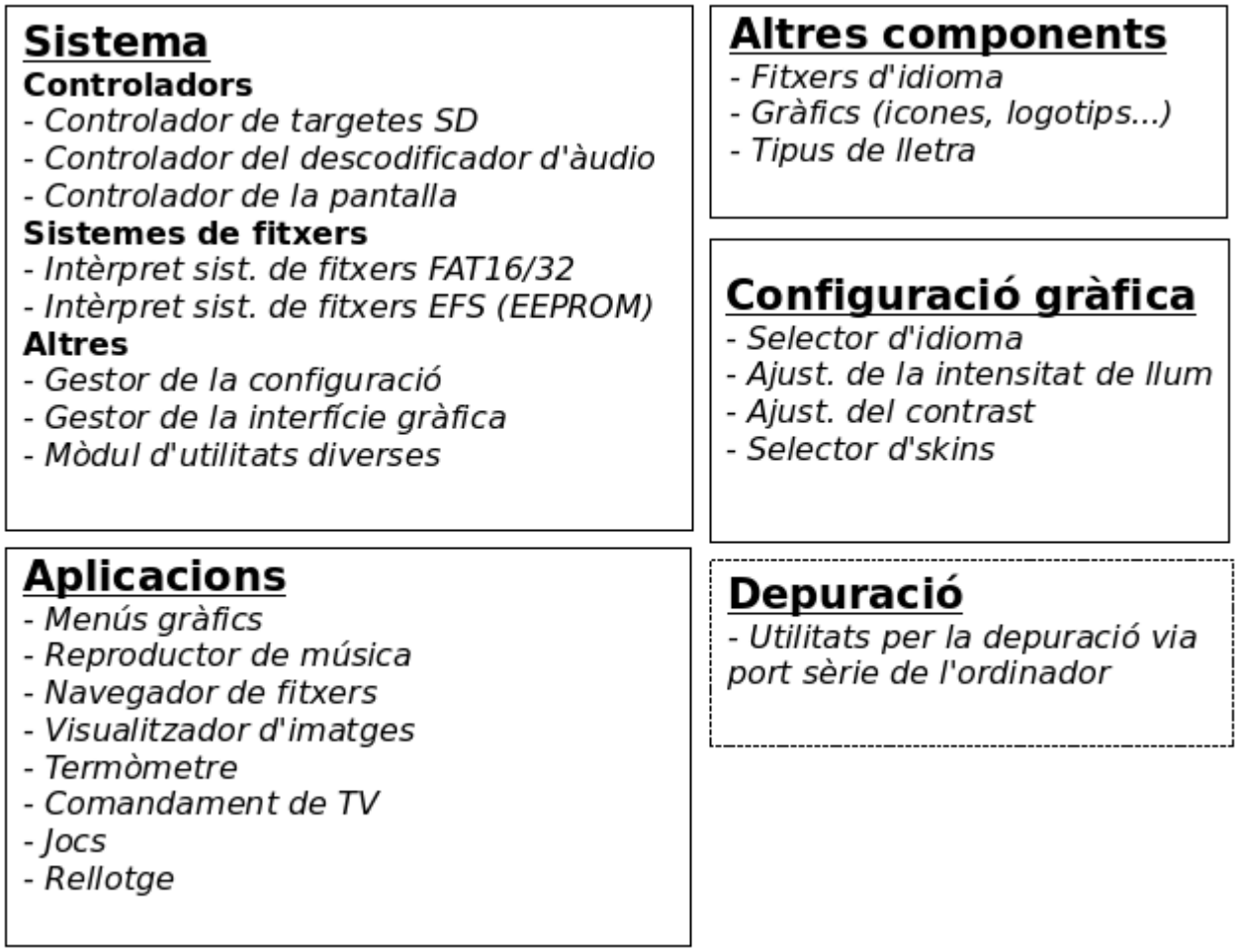
3. Estructura i funcionament del codi de l'openplayer

Tot i que en aquest document no pretenc entrar en detall en el codi de l'openplayer, penso que és important mostrar-ne la seva estructura i les diverses parts que el componen. Es pot obtenir el codi

accedint a l'Annex 5, apartat 2.1. En els següents apartats es mostraran els diversos mòduls del codi juntament amb una breu explicació de cadascun.

3.1. Estructura general del codi

He intentat classificar el codi de la manera més lògica possible en diversos apartats (Il·lustració 3.1). Tot i així internament l'estructura actual penso que és millorable en certes parts. Per exemple, no existeix una abstracció del hardware real, les aplicacions no són res més que simples funcions... és a dir, el codi és monolític, cosa que significa que qualsevol canvi o addició en el codi requereix la recompilació completa d'aquest.



Il.lustració 3.1: Estructura del codi de l'openplayer

3.2. Les parts del codi

En aquest apartat es farà una breu explicació per cadascuna de les parts, la seva importància en el reproductor i alguns altres detalls. Una documentació completa del codi es publicarà a la pàgina web oficial de l'openplayer (http://teslabs.com/openplayer), tot i que aquesta documentació quedarà fora del marge del treball per motius d'espai i també de temps.

3.2.1. Sistema

Els mòduls corresponents a l'apartat de sistema són aquells plenament essencials per l'openplayer. Alguns d'ells, com els controladors, són els encarregats d'establir comunicació amb els components electrònics i per tant és gràcies a ells qui es pot realitzar qualsevol altra tasca. Altres mòduls com l'intèrpret del sistema de fitxers FAT16/32 (el sistema de fitxers amb què treballen les targetes SD) també

es fonamental, ja que gràcies a ell es pot obtenir les cançons, imatges i tota la resta de fitxers que es troben dins la targeta SD. Finalment, també s'inclouen mòduls amb facilitats per a la gestió de la configuració, l'interfície gràfica bàsica d'usuari (icones, barra superior, etc.) com també utilitats diverses pels programadors.

3.2.2. Aplicacions

Aquest apartat el formen totes aquelles aplicacions contingudes dins l'openplayer. Potser aquest apartat és el que l'usuari observa més, ja que es treballa contínuament amb alguna d'aquestes. Les aplicacions ofereixen una interfície gràfica on l'usuari pot interaccionar com per exemple els menús, el reproductor de música... i totes les aplicacions han estat dissenyades de la manera més flexible possible. Per exemple, el navegador d'arxius és configurable i pot ser utilitzat per a qualsevol altra aplicació com a recurs per seleccionar arxius de la targeta SD (ex. navegador de música i selector d'skins, on en tots dos casos s'utilitza el navegador per a finalitats diferents). Altres aplicacions com el termòmetre permeten a altres aplicacions obtenir la temperatura actual sense necessitat d'accedir a l'aplicació en sí.

3.2.3. Configuració gràfica

L'openplayer permet ajustar certs valors des del menú de configuració. Aquest menú el componen una sèrie de petites aplicacions on cadascuna té la funció de configurar o ajustar algun valor. Un exemple és l'ajustador de la intensitat de llum o també el selector d'skins.

3.2.4. Altres components

Tot i que no són fitxers de codi sinó només de dades són essencials pel reproductor. Gràcies a ells es pot disposar de les icones, de diversos idiomes i a més dels caràcters tipogràfics que es mostren a la pantalla.

3.2.5. Depuració

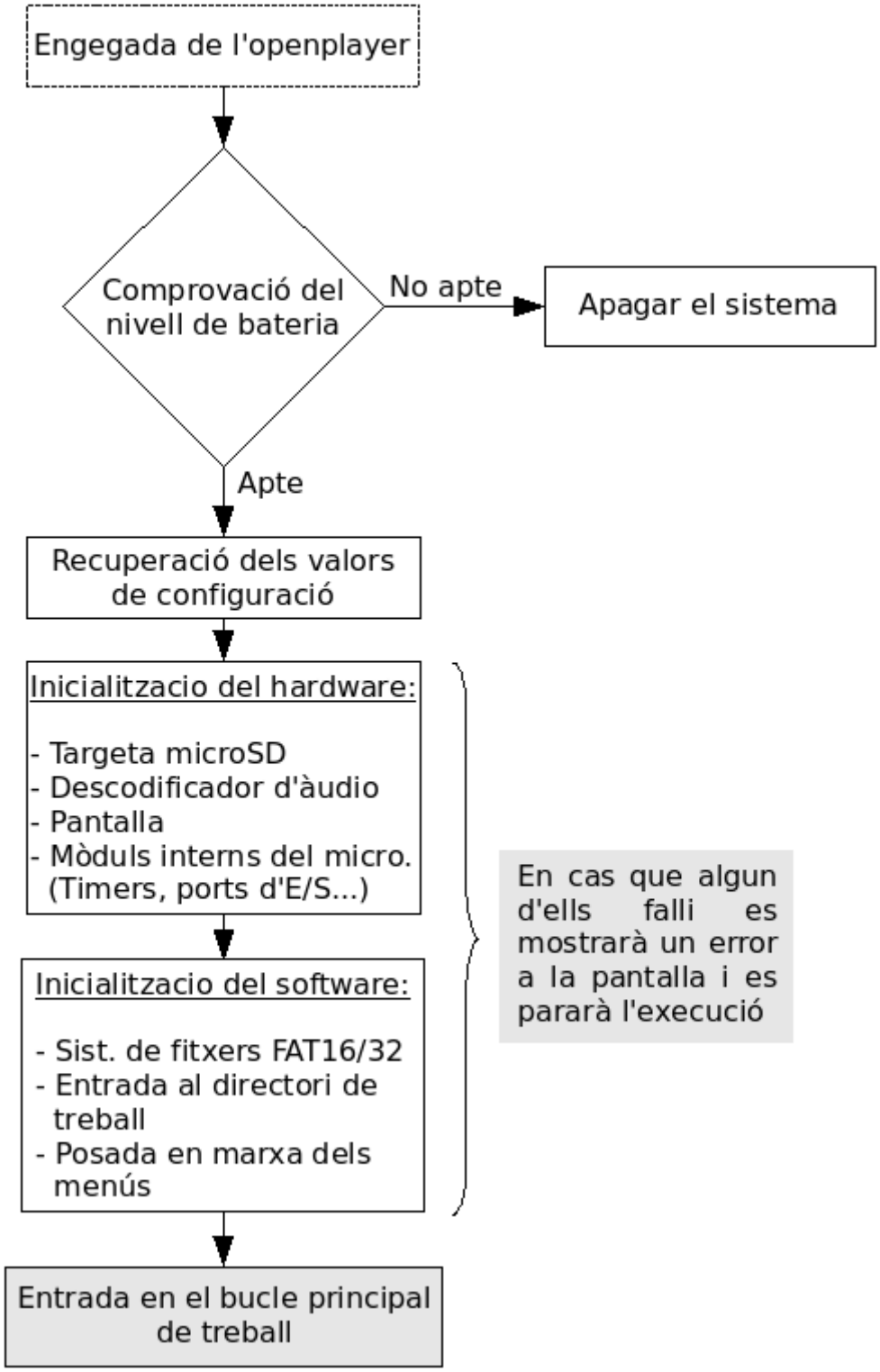
Aquest és un mòdul exclusivament per a les compilacions del firmware on es pretén depurar, és a dir, comunicar-se amb l'ordinador per resoldre possibles problemes amb el codi. En la versió final aquest mòdul és suprimit per tal de reduir el tamany del firmware i no incloure codi innecessari.

3.3. Execució del codi

A l'openplayer sempre es duu a terme alguna tasca. Abans que el reproductor estigui a punt per treballar però, s'han d'executar una sèrie de funcions que preparen els diversos mòduls del reproductor perquè es trobin operatius. Un cop fet això es posa en execució la primera aplicació que correspon a la de mostrar els menús. És per això que el primer que es veu es el menú principal en obrir l'openplayer.

3.3.1. Inicialització

Tal com havia dit, és necessari dur a terme un procés d'inicialització cada cop que s'engega l'openplayer. Els components necessiten ser inicialitzats i els mòduls obtenir els valors i la informació amb què treballar. Cal recordar que l'openplayer en realitat mai està totalment parat, excepte en aquells casos en que la bateria es trobi desconnectada o es faci un reset elèctric. El que es fa en realitat és adormir-lo i deixar-lo en mode de baix consum, de manera que queda gairebé tot parat però coses com el rellotge continuen comptant. Tot i això, cada cop que “despertem” l'openplayer aquest procés d'inicialització es duu a terme altre cop. A la Il·lustració 3.2 es pot observar un esquema amb aquest procés en detall.

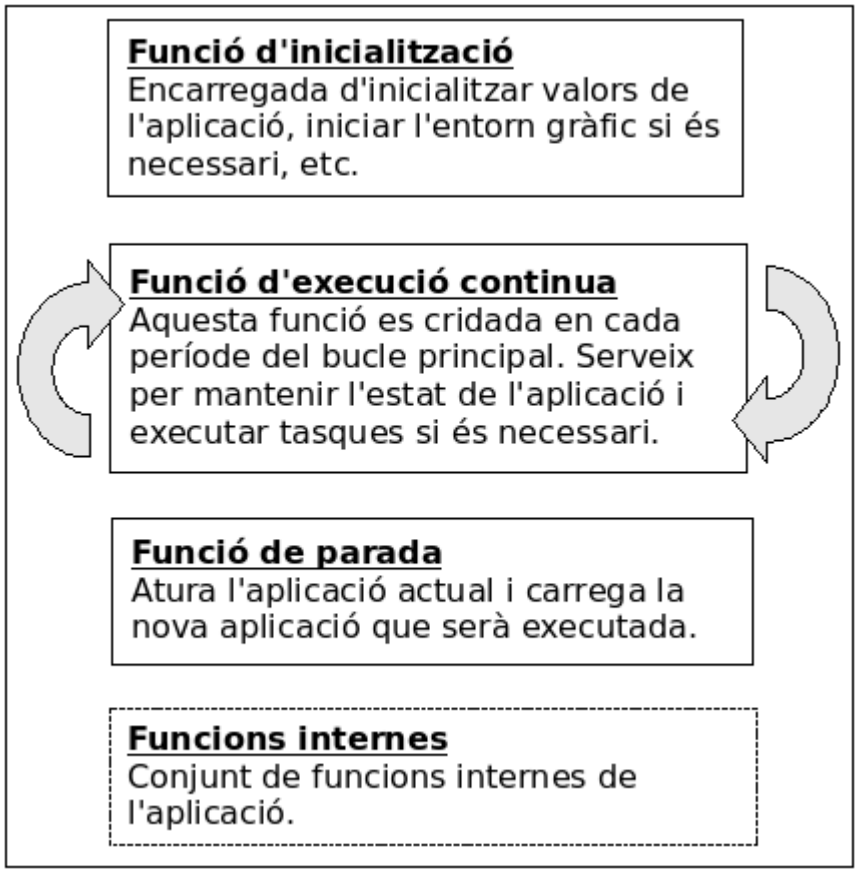


Il.lustració 3.2: Procés d'inicialització de l'openplayer

3.3.2. Estructura i execució d'aplicacions

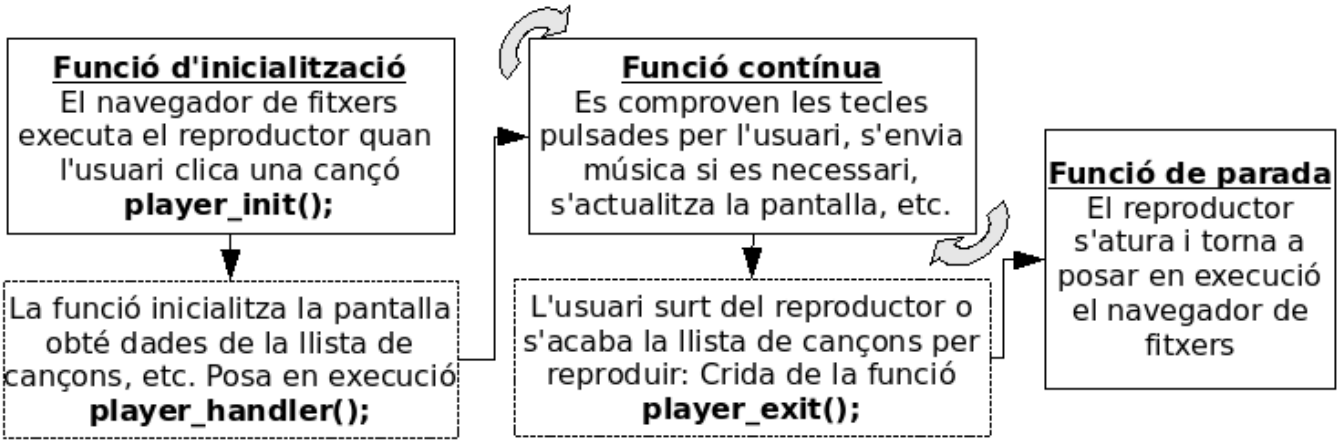
Quan es parla d'aplicacions s'està acostumat a pensar en arxius executables que es poden posar i treure en un ordinador. En el cas de l'openplayer *aplicació* pren un concepte una mica diferent ja que forma part del mateix firmware, és a dir és inseparable d'ell. En realitat una aplicació a l'openplayer no és res més que un conjunt de funcions que es troben dins un fitxer de codi. A l'openplayer tampoc existeix el *multithreading*, és a dir, l'habilitat d'executar diverses tasques en un mateix moment. Vaig descartar oferir aquesta opció bàsicament perquè la velocitat del microcontrolador és molt baixa. Per exemple, si s'està reproduint música i a la vegada s'intentés navegar pels menús seria extremadament lent. A més, requeriria obligatòriament un replantejament global del codi (ex. les esperes d'E/S o d'ús de busos de dades, cosa que

ara potser faria igualment perquè en algunes parts el codi no és massa correcte). En resum: utilitzant aquest microcontrolador és una idea inviable. El model actual també té els seus inconvenients, com per exemple la necessitat de la utilització de variables globals o estàtiques que resideixen constantment a la memòria, tot i que de moment no han suposat cap problema d'espai. L'estructura que vaig escollir finalment per les aplicacions es mostra a la Il·lustració 3.3.



Il.lustració 3.3: Estructura d'una aplicació de l'openplayer

A la Il·lustració 3.4 es pot observar com és el procés d'execució d'una aplicació mitjançant l'exemple del reproductor de música de l'openplayer.



Il.lustració 3.4: Exemple de l'aplicació del reproductor de música de l'openplayer

El model actual potser no és la opció més eficient. Vaig decidir crear aquest model perquè permetia l'execució d'aplicacions successives podent aturar l'anterior abans d'executar la segona i mantenint l'estat ja que les variables eren globals. Sincerament ara el veig un model molt precari i una mica absurd en un

reproductor de música, però ja es massa tard per corregir-lo i entregar aquest treball a temps. La veritat és que alguns models de *multithreading* fets per AVR m'atreien molt més, ja que evitaven tenir valors estàtics a la memòria si una aplicació no s'executava, la prioritat d'execució... El problema ha estat sempre la velocitat dels microcontroladors AVR en l'openplayer on es treballa tant ajustat. El cas és que serà un factor que em plantejaré més seriosament en algun futur projecte, inclosa la selecció d'altres microcontroladors o utilitzar com a base un petit sistema operatiu per a microcontroladors.

Capítol 3

Evolució

1. Un procés llarg

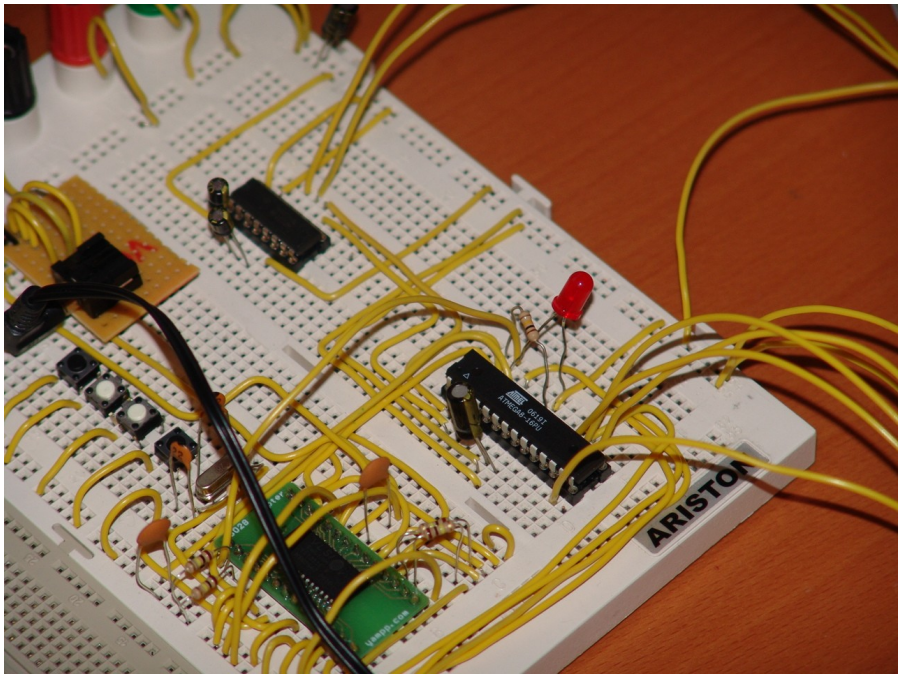
L'openplayer ha passat per diverses fases d'estat durant la seva construcció i programació. L'aspecte actual és tant sols una petita part d'aquest llarg procés. Durant aquest procés, vaig anar fent fotografies i recol·lectant dades perquè ara es puguin observar els canvis i també fer comparacions. Els canvis en el firmware han estat constants durant el procés de construcció, i de fet, es molt probable que continuïn durant els pròxims mesos per arreglar possibles errors o afegir alguna funcionalitat nova. Altres parts com el hardware també han evolucionat tot i que lògicament a menys velocitat. En els següents punts es mostrarà un resum d'aquesta evolució juntament amb algunes imatges[4].

2. Evolució del hardware

Els canvis en el hardware han estat gairebé sempre per afegir algun component nou tot i que en algun cas com es veurà a continuació ha estat per canviar a una versió millor d'algun component. L'addició d'un nou component és sempre una tasca difícil i és el moment en el què tens més dubtes de tots.

2.1. L'àrea de treball

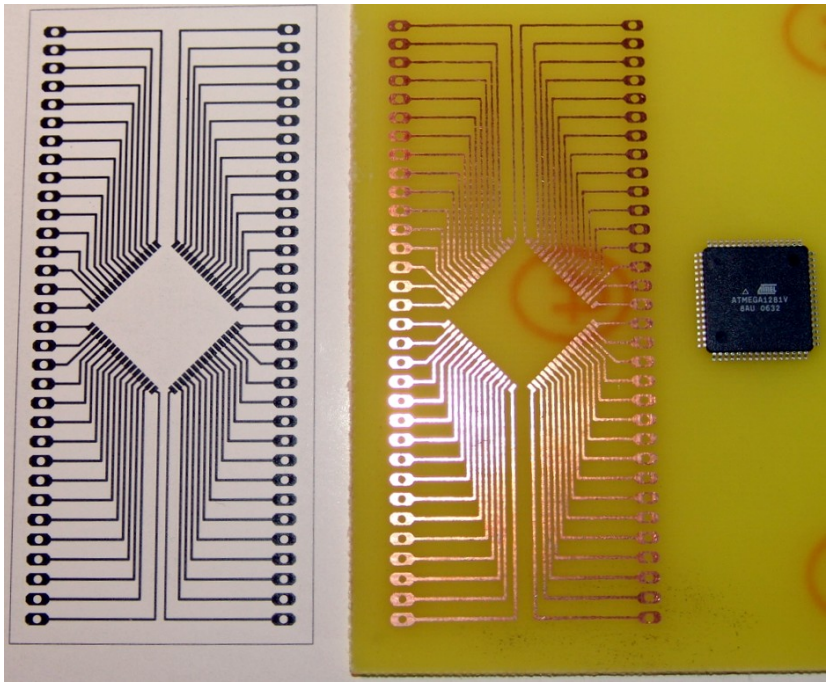
L'àrea de treball abans de fabricar la placa actual era una protoboard on totes les connexions es feien mitjançant cables (Il·lustració 2.1). És un mètode rudimentari però que en aquest cas ha sigut prou factible i no ha comportat problemes d'estabilitat.



Il.lustració 2.1: Un dels primers muntatges de l'openplayer

Tot i això alguns xips no es fabriquen amb els encaixos d'una protoboard i per tant, en aquests casos va caldre fer o comprar plaques extensives que actuessin com a adaptador. Les primeres van ser les del descodificador d'àudio les quals vaig comprar ja que llavors encara no coneixia com fer-les jo mateix. Més endavant però, vaig necessitar més plaques d'adaptació i vaig trobar-me amb alguns manuals que explicaven com fabricar-les mitjançant un mètode casolà. La qualitat era molt bona, així que vaig decidir aprendre'n. No va ser fàcil obtenir la primera (la del microcontrolador actual) ja que les patilles eren de menys de mig mil·límetre. En el moment que vaig obtenir la placa mostrada a la Il·lustració 2.2 em vaig plantejar seriosament considerar la sort com a causa d'aquell resultat.

4 Podeu veure la col·lecció de fotografies completa accedint a l'Annex 4



Il.lustració 2.2: Placa d'adaptació del microcontrolador actual

El mètode de fabricació de plaques de circuit imprès utilitzat es descriu en detall a l'Annex 2.

2.2. El microcontrolador

El primer microcontrolador de l'openplayer no era el mateix que l'actual (Atmega1281), sinó que era un altre model de la sèrie AVR amb menys funcionalitats, concretament l'Atmega8. Vaig començar a treballar amb aquest altre microcontrolador perquè no requeria soldar-se, sinó que podia posar-se directament a la protoboard. Això era un avantatge per a un soldador sense experiència com era llavors. Les seves capacitats en comparació a l'actual són molt inferiors (vegeu Taula 1), tot i que la velocitat de càlcul no varia (8MIPS).

Il.lustració 2.3: Atmega8

Característica	Atmega8 (PDIP-28)	Atmega1281 (TQFP64)
Memòria Flash	8KB	128KB
Memòria EEPROM	512Bytes	4KB
Memòria SRAM	1KB	8KB
Timers	2 de 8bits i 1 de 16bits	2 de 8bits i 4 de 16bits
JTAG	No disponible	Disponible
Nombre d'E/S	23	54

Taula 1: Diferències més importants entre l'Atmega8 i l'Atmega1281

En les primeres versions del codi, l'Atmega8 era suficient ja que les tasques que es duien a terme feien poc ús de la memòria i el codi s'estenia a poc més d'un controlador de la targeta SD, un intèrpret de sistema de fitxers FAT16/FAT32 i un reproductor de música molt bàsic (Il·lustració 2.1). En el moment que vaig introduir la pantalla al circuit (Juny de 2007) vaig decidir canviar al microcontrolador actual ja que les capacitats s'havien quedat petites i necessitava poder continuar endavant amb la programació del

firmware. Vaig escollir l'Atmega1281 perquè era el model que oferia més memòria SRAM de la sèrie AVR. Actualment no s'utilitza aquesta quantitat de memòria i amb un microcontrolador de característiques inferiors (ex. Atmega128) seria possible executar el firmware de l'openplayer sense problemes. Tot i així vaig voler estalviar-me possibles mancances seleccionant aquest (una mala pràctica si es pretengués comercialitzar l'openplayer).

2.3. El descodificador

El model del descodificador també ha canviat tot i que en principi només ho vaig fer per qüestions d'espai. Finalment el canvi m'ha permès a part de reduir el tamany de la placa, oferir la compatibilitat amb nous formats de música. La veritat és que aquest canvi m'ha suposat un problema en el disseny del circuit final, ja que no vaig tenir en compte alguns canvis importants que requerien modificacions en el circuit i en la primera revisió de la placa és inestable si no es fan algunes correccions. Per aquest motiu vaig decidir arreglar aquest error i fabricar una nova versió de la placa a última hora on queda corregit, i a més s'ha permès incloure suport per una versió encara més nova del descodificador d'àudio que l'empresa VLSI ha llençat a finals d'aquest any 2007.

2.3.1. VS1011e

El primer model va ser el descodificador VS1011e el qual vaig seleccionar perquè era l'únic model que es trobada en un tamany que permetia soldar-lo fàcilment a la botiga on el vaig comprar. Aquesta versió oferia descodificació dels formats MP3, MIDI i WAV, suficients pels meus propòsits. La veritat és que ha estat un descodificador que no ha donat gaires problemes i la seva implementació ha estat fàcil. L'únic problema que té es el volum màxim en trobar-se connectat a uns auriculars, que no és massa alt. Aquesta versió del descodificador amb la versió de tamany petit es pot utilitzat a la revisió 1.0 del circuit per obtenir un funcionament estable del reproductor.

Il.lustració 2.4: VS1011e

2.3.2. VS1033

En el moment de realitzar el disseny final vaig decidir canviar a una versió reduïda del descodificador d'àudio. En aquell moment vaig observar que havien aparegut nous models que oferien nous formats a més de noves característiques i al mateix preu que les versions anteriors. L'últim era el model VS1033, que a més dels formats descrits anteriorment afegia suport per WMA i AAC. Aquesta versió també és capaç de gravar veu provinent d'un micròfon tot i que a l'openplayer no s'utilitza aquesta funció. La versió VS1033 és compatible amb la revisió 1.1 del circuit.

2.3.3. VS1053

Just a últim moment, vaig necessitar fer una sèrie de revisions a la placa per corregir alguns errors (revisió 1.1). En aquell mateix moment l'empresa VLSI acabava de posar a disposició del públic el xip VS1053, que incorporava el suport pel format OGG a més de tots els anteriors. Això em va cridar molt l'atenció ja que el format OGG és un format de caràcter lliure, i juntament amb la filosofia de l'openplayer podien fer un bon conjunt, així que no vaig dubtar en comprar-ne una mostra i fer la nova revisió del circuit compatible amb aquesta nova versió. Tot i això, la inclusió de nous formats requereix algunes modificacions de software per oferir-ne un suport complet que en la versió entregada en aquest treball (Gener de 2008) no estan disponibles. En noves versions del firmware publicades a la pàgina web oficial quedaran reflectits aquests canvis.

2.4. La targeta SD

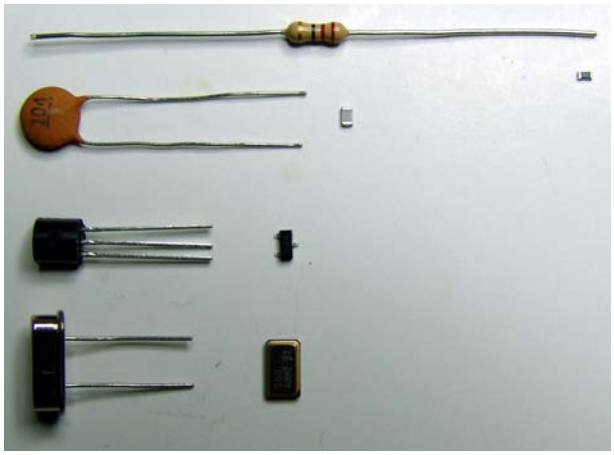
A la versió final de l'openplayer s'utilitzen les microSD en comptes de les SD per motius d'espai. En realitat el funcionament de les microSD és idèntic al de les SD. Tot i així, vaig haver de reescriure el controlador d'aquestes targetes en el moment que vaig fer el canvi ja que vaig poder comprovar que

algunes targetes fallaven. De fet, era un error meu per no complir amb exactitud les pautes marcades pel protocol que s'estableix en les especificacions de les targetes SD, així que un cop corregit (no va ser una tasca fàcil) tot va quedar arreglat. A més, la reescriptura del controlador també va suposar el suport per les noves SD d'alta capacitat (*SDHC*). A la Il·lustració 2.5 es pot apreciar el guany d'espai gràcies al canvi a les targetes microSD.

Il.lustració 2.5: Comparació de tamany entre una SD i una microSD

2.5. El canvi a components superficials

Durant el procés de construcció inicial, a la protoboard la majoria de components passius (resistències, condensadors...) que s'utilitzaven eren muntables directament en els mateixos forats i per tant no calia soldar-los. Aquests components però eren d'un tamany molt més gran que no pas els actuals. A més, quan aquests components són muntats en una placa de circuit imprès, necessiten ser soldats a través d'un forat, cosa que significa que ocupa espai a les dues cares[5]. Solen ser utilitzats en circuits de més potència ja que el fet de ser més grans també implica que puguin resistir el pas de més intensitat per a un mateix voltatge. També són utilitzats freqüentment per a prototips fets a casa (com en aquest cas) ja que són molt còmodes de soldar i manejar. Evidentment però, si pretenia reduir el reproductor a un tamany de butxaca aquests components eren inviables i per tant vaig haver de seleccionar un nou tipus de components, els SMT (*Surface Mount Technology*) que són soldats superficialment.



Il.lustració 2.6: Components de muntatge amb forat juntament amb el seu equivalent SMT

En alguns components no passius, com el sensor de temperatura, també es va fer aquest canvi però això no implica en res el seu funcionament.

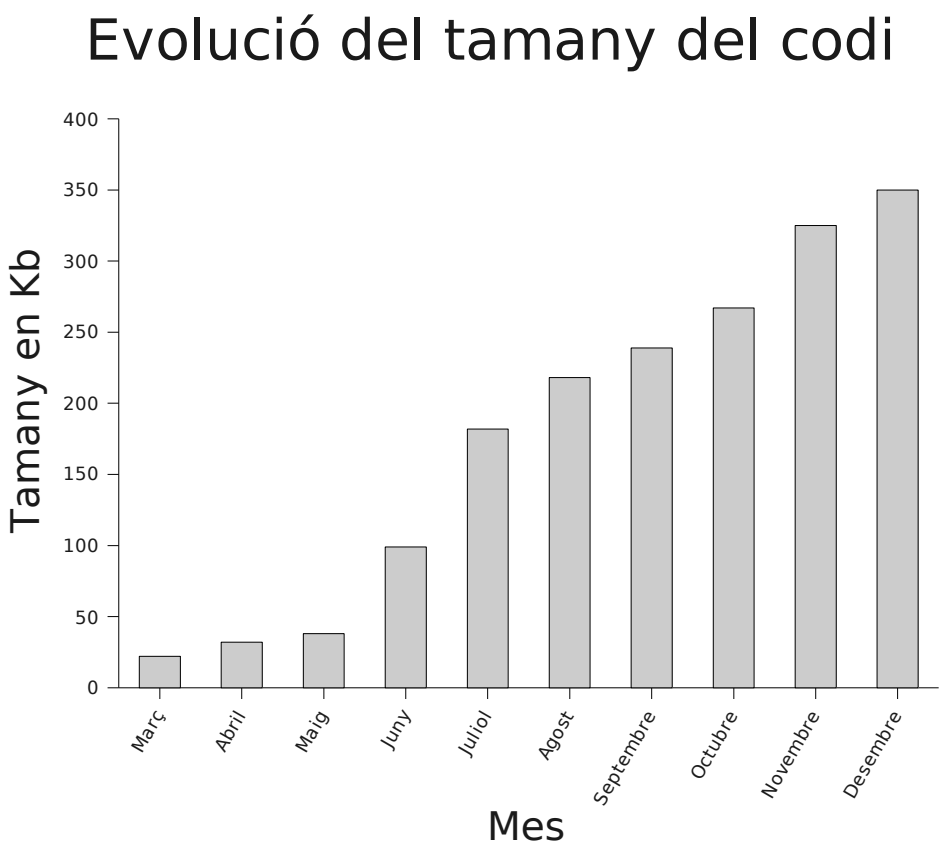
5 Vegeu Capítol 4, apartat 2.2

3. Evolució del firmware

Tal com deia, el firmware ha estat una de les parts més canviants de l'openplayer i probablement també s'acabi de polir en els pròxims mesos. Durant el procés de construcció vaig anar desant còpies del firmware per poder fer-ne comparacions al final. No les faré públiques per evitar situacions de posada en evidència, que de fet són totalment segures en el codi actual.

3.1. Estadístiques de progrés del codi

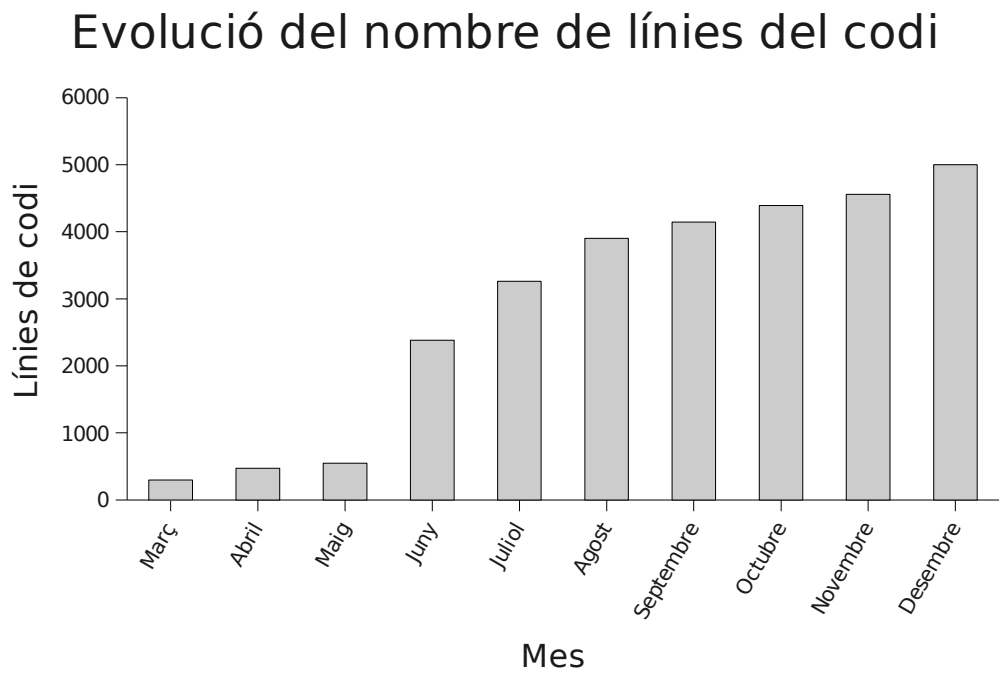
Tot i que les estadístiques següents són un mètode poc fiable d'avaluar qualsevol codi si que poden ajudar a poder quantificar el progrés. En el cas del tamany he contat el codi al complet, és a dir, s'inclouen el conjunt de dades com els gràfics o idiomes que estan desats en fitxers de codi. Per tant, el tamany del que podem considerar purament codi és en realitat inferior. En el Gràfic 1 podem observar l'evolució del tamany, on també es pot comprovar com en els mesos d'estiu és on es realitza la major part del progrés i com aquest s'estabilitza al final, que és quan vaig dedicar més temps en el disseny del circuit imprès. Els primers mesos també es pot veure un progrés molt baix en relació als altres, ja que va ser un període d'adaptació de treball als microcontroladors AVR com també la dificultat d'obtenir els primers muntatges estables on poder treballar.



Gràfic 1: Evolució del tamany del codi

També he pensat que podia ser interessant posar el gràfic amb l'evolució del nombre de línies, que de fet molts cops pot ser un indicador molt relatiu, ja que no significa que un major nombre de línies equivalguin a un bon codi. En tot cas podríem avaluar la velocitat d'execució respecte altres codis que facin la mateixa funció, però per això necessitaríem un altre codi amb què comparar. La mesura s'ha fet amb el programa *cccc*, el qual realitza una estadística completa del codi tot i que aquí només se'n mostra una petita part. El nombre de línies mesurat descompta les línies en blanc, comentaris, definicions i altres parts que no són en realitat codi funcional. Podeu consultar a la web del *cccc* (http://cccc.sourceforge.net)

per obtenir més informació sobre la metodologia utilitzada per obtenir aquesta estadística.



Gràfic 2: Evolució del nombre de línies del codi

3.2. Tasques no acabades

M'hauria agradat poder incloure altres funcions a l'openplayer però malgrat l'intent no ha estat possible. Primerament, el suport USB que hauria permès poder connectar l'openplayer a l'ordinador i poder transferir-hi cançons directament que vaig aconseguir fer funcionar però per algun motiu que encara desconec era molt inestable. Es muntava el dispositiu a l'ordinador però al cap de poc es desconnectava automàticament. La falta de temps em va impedir poder seguir investigant l'error ja que s'havien d'acabar altres parts del codi. Tot i així hauria estat un suport USB amb velocitats de transferència baixíssimes, ja que les dades han de passar a través del microcontrolador i aquest és molt lent per poder oferir les velocitats corresponents a les de l'USB 1.1/2.0. Una de les altres opcions que volia incloure era un emissor FM, que hauria permès sintonitzar en qualsevol ràdio el que s'estava escoltant en aquell instant a l'openplayer. Vaig intentar-ho amb un nou xip que permetia l'ajustatge de la freqüència digitalment tot i que tampoc va funcionar bé. El factor temps ha estat un dels grans problemes a l'hora de voler afegir noves funcionalitats, ja que com havia mencionat anteriorment requereix moltes hores l'aprenentatge del funcionament d'un nou xip (ex. l'USB on s'havia d'aprendre el funcionament del protocol).

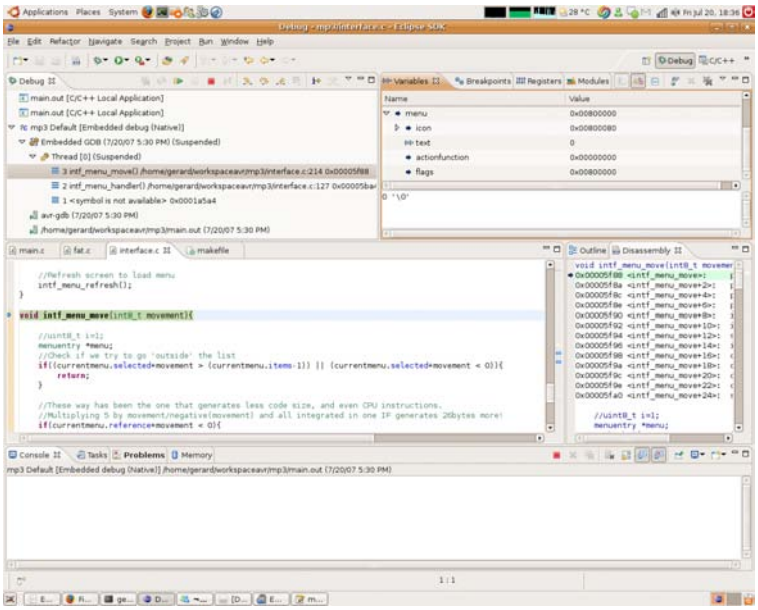
3.3. Millores en els mètodes de programació

Al principi no vaig decidir una estructura estricta a l'hora d'escriure el codi, tot i que l'he anat definint a mesura que he anat avançant per tal d'oferir un codi més ordenat i polit. Això inclou la metodologia de nominació de funcions, noms de tipus de variables, el mode d'escriptura en general, etc.

Il.lustració 3.1: JTAGICE-mkII

A mesura que el codi va creixent és comú trobar-se amb errors programats anteriorment que poden afectar a un codi nou que has escrit, i molts cops trobar-los és una tasca molt dura. Per aquest motiu, des d'un bon principi vaig crear un petit mòdul de depuració que es connectava a l'ordinador mitjançant el port sèrie i permetia enviar valors de posicions de memòria, texts d'avís en arribar en un punt determinat, etc. És un mètode en alguns casos una mica rudimentari i vaig acabar descobrir la disponibilitat d'un dispositiu fabricat per *Atmel*, el *JTAGICE-mkII* (Il·lustració 3.1), que permet la depuració en temps real. Això significa poder seguir l'execució del

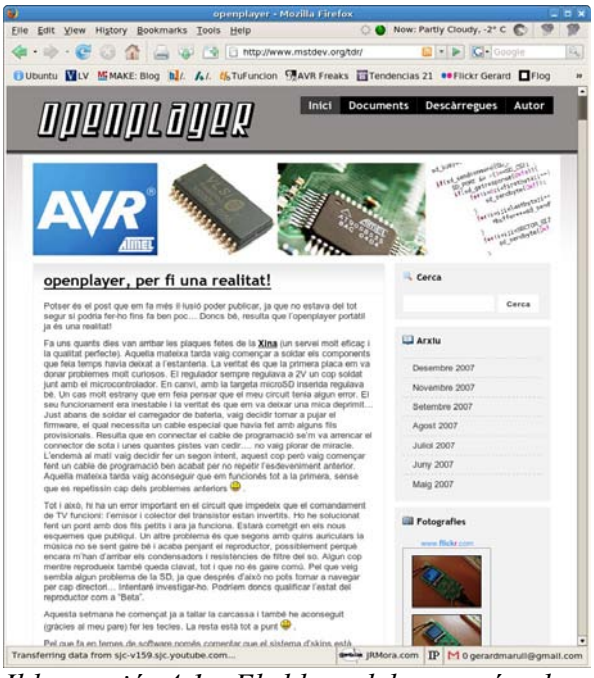
programa, parar-la en moments determinats per tal de comprovar l'estat del microcontrolador (valors de variables, registres del microcontrolador...), és a dir, permet fer una depuració del firmware de la mateixa manera que ho faríem en un programa d'ordinador convencional. Aquest mateix dispositiu també permet programar el microcontrolador, configurar-lo, etc. La veritat és que va ser una inversió cara però no m'arrepenteixo gens d'haver-lo obtingut.



Il.lustració 3.2: Depuració en temps real mitjançant el JTAGICE-mkII, l'avarice i l'entorn Eclipse

4. El blog de l'openplayer

Tot i que no ha estat gaire visitat, vaig crear un petit blog on he anat publicant periòdicament l'estat del projecte, les idees que han anat sorgint... Penso que ha estat un bon mètode per oferir el seguiment del projecte i també per constatar-ne l'autenticitat. Podeu visitar-lo accedint a http://mstdev.org/tdr.



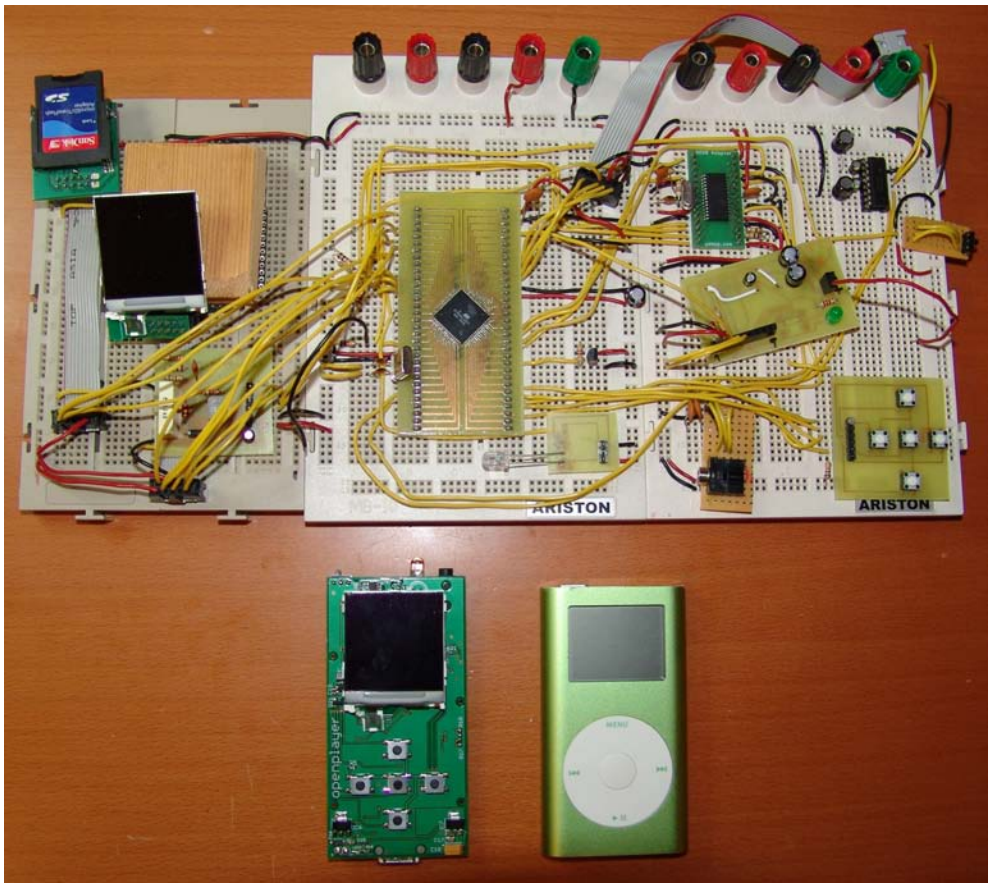
Il.lustració 4.1: El blog del progrés de l'openplayer

Capítol 4

Disseny i construcció

1. La versió de butxaca

Quan vaig començar amb l'openplayer, la meva idea era fer-lo de butxaca ja que no tenia massa sentit fer-lo només en una protoboard gran amb un aspecte llunyà al d'un reproductor comercial. Tot i això és difícil imaginar-se com comprimir una protoboard tant gran en una placa tant petita quan no s'ha fet mai aquest procés. És per això que aquest pas era un dels més temuts quan vaig començar amb el projecte, doncs no tenia experiència en el món del disseny de circuits impresos. El tamany de l'openplayer ha canviat radicalment en aquest procés. Podeu observar-ho a la Il·lustració 1.1, on es mostren oposades les dues versions del reproductor i a més, s'hi ha afegit un *iPod mini* (any 2005) per poder comparar-lo amb un reproductor de música comercial. En els pròxims capítols s'explicarà en detall tot aquest procés.



Il.lustració 1.1: Comparació entre la versió gran i petita de l'openplayer juntament amb un iPod mini de l'any 2005

2. Selecció de components

La selecció dels components finals de l'openplayer no ha estat fàcil ja que hi han intervingut diversos factors: disponibilitat, varietat, preu... Sortosament però, he pogut tenir accés a bons proveïdors que han facilitat el procés d'obtenció.

2.1. Proveïdors de components

A la primera versió de l'openplayer la majoria de components són molt comuns i obtenir-los ha estat fàcil, ja que es poden comprar en qualsevol botiga d'electrònica convencional, com per exemple el *Zeners* de la ciutat de Girona. Els xips però, són uns elements molt específics i no es troben en qualsevol lloc. Vaig cercar la disponibilitat de botigues a Espanya per alguns dels components però com era d'esperar no hi va haver sort. Alguns d'ells com la pantalla, el descodificador o el primer microcontrolador els vaig

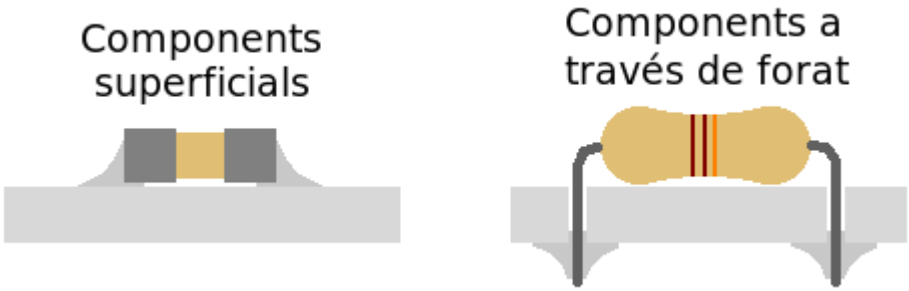
poder comprar aquí a Europa, concretament a la botiga www.jelu.se. En el moment que vaig necessitar comprar el nou microcontrolador vaig haver de cercar alguna botiga més específica. A través de la pàgina web d'*Atmel* vaig conèixer la botiga *Digikey* (www.digikey.com) localitzada als Estats Units. Realment, va ésser un gran descobriment i de fet, a la versió final de l'openplayer gairebé la totalitat dels components hi són comprats. *Digikey* disposa d'un catàleg de més de 2.200 pàgines i ofereix vendes al detall i també a l'engròs. Un dels altres factors que m'ha afavorit a l'hora de comprar a *Digikey* ha estat la caiguda del dòlar nord-americà durant els últims mesos.

Un dels altres mètodes d'obtenció de components ha estat mitjançant la petició de mostres gratuïtes. Algunes empreses d'on les he obtingut han estat *Maxim* i *Linear Technology*. La quantitat enviada és petita (normalment de dues unitats) però suficient per prototips.

Il.lustració 2.1: Alguns dels proveïdors

2.2. Avantatges dels components superficials

En el capítol anterior[6] es descrivia el canvi a components superficials. L'ús d'aquests components comporta un gran avantatge en llocs on l'espai és un factor important. Un component soldat a través de forats inhabilita l'espai corresponent al forat a totes les capes que es troben sota seu, en canvi en un de superficial això no passa, ja que se solda a la superfície. A més, els components superficials s'ofereixen en una gamma de tamanys molt reduïts. A la Il·lustració 2.2 es pot observar la diferència entre els dos tipus de components soldats en una placa de circuit imprès.



Il.lustració 2.2: Comparació entre components superficials i de forat

2.3. Compromís amb el medi ambient: compliment del RoHS

L'any 2003 la Unió Europea va adoptar una nova directiva anomenada RoHS (*Restriction of Hazardous Substances*). Aquesta directiva va entrar en vigor l'1 de juny de 2006 i restringeix l'ús de sis substàncies perilloses en la fabricació de certs equips electrònics, que concretament són:

- Plom
- Mercuri
- Cadmi
- Crom VI
- PBB (*PolyBrominated Biphenyls*)
- PBDE (*PolyBrominated Diphenyl Ether*)

6 Vegeu Capítol 3, apartat 2.5

La directiva RoHS afecta a una gran gamma de productes amb contingut electrònic com els electrodomèstics, electrònica de consum, joguines, etc. En altres paisos també s'han creat directives semblants basades en aquest model, cosa que ha fet que actualment una gran quantitat de components electrònics siguin oferts sota el compliment d'aquesta directiva. Podreu observar una etiqueta (Il·lustració 2.3) en molts productes electrònics, la qual suposa el compliment de la normativa i que aquest no conté cap substància de les descrites anteriorment.

Il.lustració 2.3: Exemple del compliment RoHS en un producte informàtic

Quan vaig seleccionar els components de l'openplayer final vaig tenir molt en compte aquest factor. Alguns proveïdors com *Digikey* estan totalment adaptats a aquesta nova directiva i a l'hora de comprar components pots incloure en els resultats de cerca únicament aquells que la compleixen. Altres components com la pantalla, que vaig obtenir a una altra botiga (*Jelu handelsbolag*), no tenen la documentació que ho acrediti tot i que els propietaris de la botiga em van confirmar que complia el RoHS. Finalment, la placa de circuit imprès també ha estat fabricada amb el compliment d'aquesta norma, ja que el fabricant ofereix aquesta possibilitat en el procés d'acabat.

Evidentment per poder acreditar que un producte compleix la normativa RoHS requereix una inspecció i documentació completa del projecte. Com que l'openplayer no pretén ser cap reproductor comercialitzable és innecessari tot aquest procés i el compliment del RoHS és només una dada complementària.

3. El disseny i fabricació del circuit imprès

El disseny del circuit imprès final ha estat una de les parts més importants de l'openplayer, després de la programació del firmware. En els següents punts s'explicarà què són els circuits impresos, com es dissenyen i què s'ha fet per fabricar-lo.

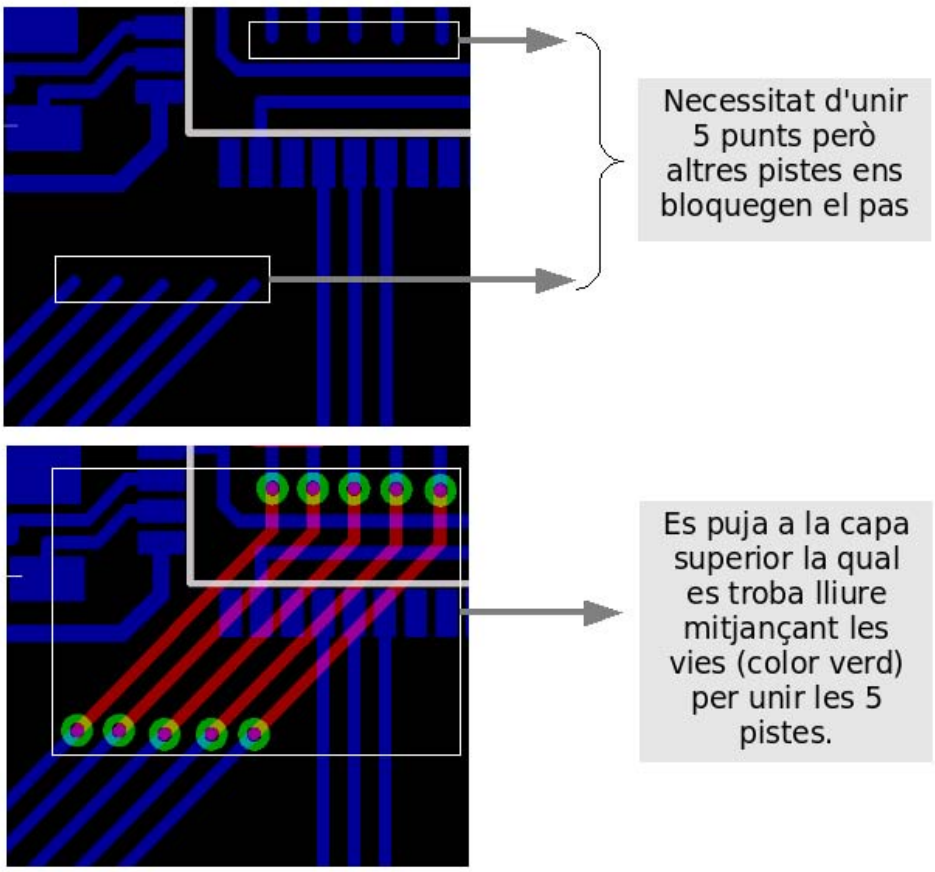
3.1. Els circuits impresos

Els circuits impresos són objectes molt comuns en la nostra vida quotidiana i podem trobar-ne en qualsevol aparell on es requereix col·locar components electrònics (xips, resistències, LEDs....). Són utilitzats per subjectar i connectar elèctricament els components d'un circuit. La seva superfície consisteix en una làmina fina de coure que pren la forma del circuit que s'ha dissenyat i després revelat mitjançant

diversos processos químics. El circuits impresos poden prendre una alta complexitat, ja que no només poden ser d'una sola capa, sinó que poden ser multicapa. L'empresa a la qual he fet fabricar la placa de l'openplayer disposa de tecnologia suficient per fabricar plaques de fins a 24 capes. En el cas de l'openplayer, és un circuit relativament simple i que només està format per les dues capes visibles (superior i inferior).

Il.lustració 3.1: Exemple d'un circuit imprès

L'ús de diverses capes es deu al fet que molts cops no es disposa de suficient espai a la capa actual per poder continuar la ruta. Si a la part inferior d'aquella zona no hi ha cap component ni tampoc cap pista es pot *baixar* a aquesta altra capa per travessar la pista que ens impedeix el pas i tornar a pujar altre cop per continuar la ruta. Alguns cops és un procés complicat ja que cal repetir-lo més d'un cop i fins i tot fer petits desplaçaments per trobar espais lliures. Aquest canvi de capa per poder continuar la ruta, implica la creació de petits forats anomenats *vies* els quals són conductors i uneixen els dos costats. A la Il·lustració 3.2 es mostra un exemple d'aquest procés.



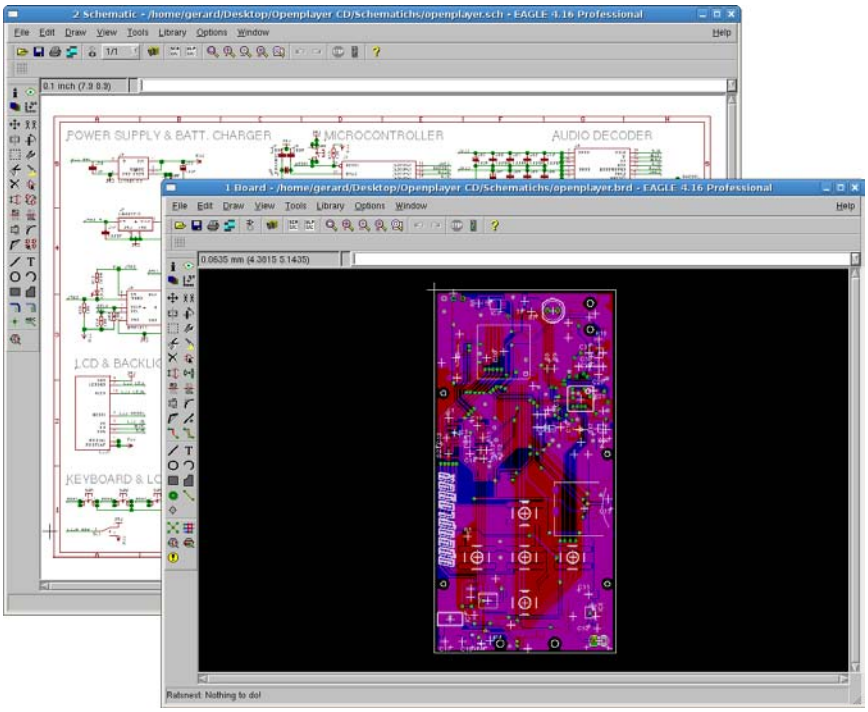
Il.lustració 3.2: Mètode de creació de circuits de doble capa

3.2. El procés de disseny

El procés de disseny és un procés complex i requereix moltes hores (també paciència). Mitjançant els ordinadors actuals aquest procés se simplifica però tot i això continua sent un procés que no és fàcil com es veurà a continuació.

3.2.1. El programa utilitzat

No vaig poder trobar un programa lliure de qualitat relacionat amb el disseny de circuits, així que vaig decantar-me per la solució comercial *Eagle*, un programa especial per aquesta tasca i que a més ofereix versió per Linux, MacOS X i Windows. El programa és molt complet i amb ell he sigut capaç de realitzar totes les tasques que se m'han presentat durant el procés de disseny. Tot i així crec que és millorable en certs aspectes.



Il.lustració 3.3: Programa de disseny de circuits Eagle

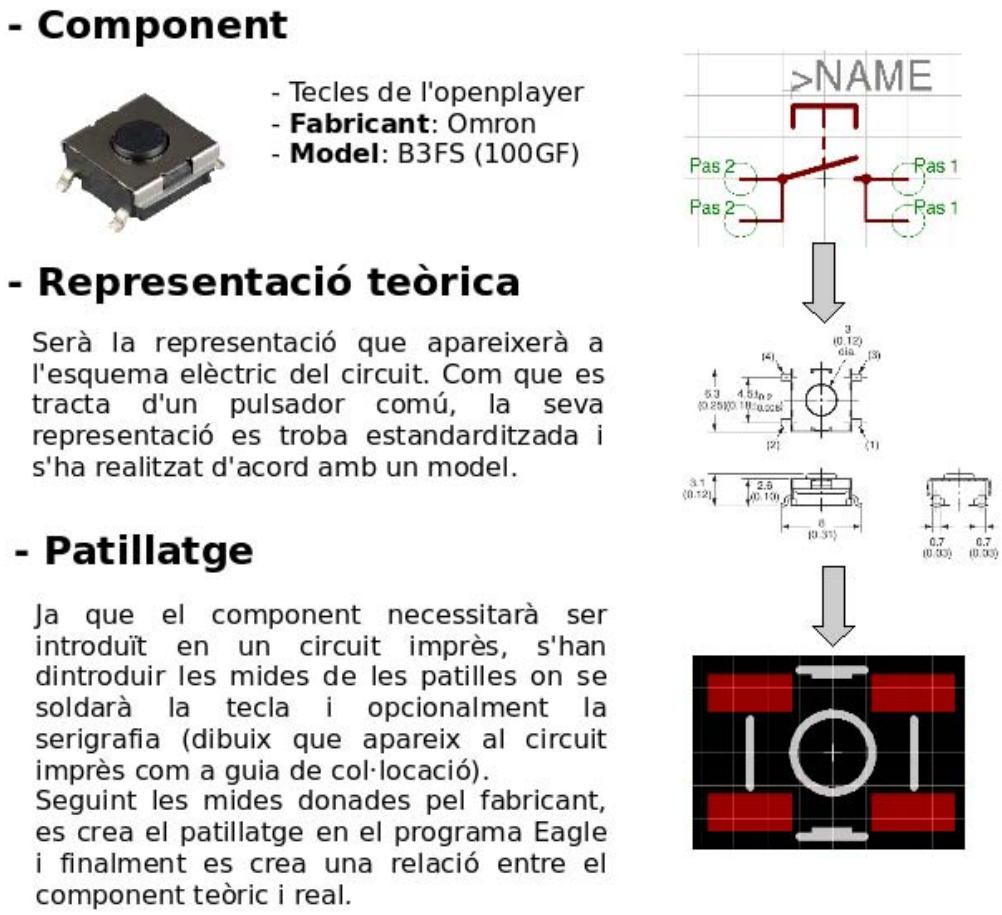
3.2.2. L'esquema elèctric

Abans de poder començar el disseny del circuit imprès, és necessari realitzar l'esquema elèctric o teòric del circuit, on es poden observar clarament tots els components del circuit, el seu nom, els seus valors si és necessari (ex. resistències) i també la relació entre ells (connexions). A partir d'aquest esquema, podrem començar a dissenyar el circuit imprès, però no abans[7].

3.2.3. Llibreria de components

Tan a l'esquema elèctric com en el circuit imprès, apareixen els components que formen l'openplayer. La majoria d'aquests components es troben ja fets a la col·lecció de llibreries que incorpora l'Eagle, o en alguns casos són descarregables a internet d'altra gent que els ha fet. En altres casos però, aquests components no existeixen en cap llibreria i cal crear-ne una de nova. En el cas de l'openplayer ha calgut fer-ho amb diversos components. Aquest procés comporta dues parts: la realització de la representació teòrica del component i el seu patillatge en una placa de circuit imprès. A la Il·lustració 3.4 es pot observar un cas real de l'openplayer.

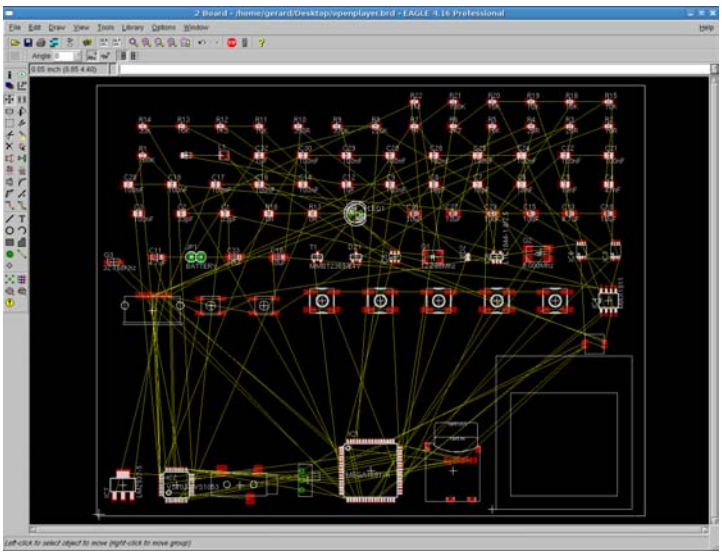
7 Es pot trobar l'esquema elèctric de l'openplayer accedint a l'Annex 1



Il.lustració 3.4: Procés de creació d'una llibreria amb un component

3.2.4. El circuit imprès

Aquest és el procés més complicat com he repetit diversos cops. En primer lloc, cal una selecció de la posició de components òptima per tal d'aprofitar al màxim el poc espai del que disposem, ja que en el moment que passem el nostre esquema elèctric a circuit imprès, ens trobarem una imatge com la de la Il·lustració 3.5.



Il.lustració 3.5: Circuit encara no ordenat ni connectat

El procés d'ordenació és lògicament manual, i el connexionat pot realitzar-se manual o automàtic amb la funció inclosa en el programa Eagle anomenada *Autorouter*, tot i que en algunes zones del circuit és recomanable no utilitzar-lo ja que requereixen un traçat especial per evitar problemes d'interferències.

El resultat de tot aquest procés en el cas de l'openplayer es pot observar a l'Annex 1 on es troben les imatges del circuit imprès de l'openplayer.

3.3. El procés de fabricació

Anteriorment havia mencionat la creació de circuit impresos mitjançant un mètode casolà. A la versió final del reproductor però, es requerien pistes molt petites (ex. descodificador d'àudio) i era pràcticament impossible obtenir un bon resultat amb aquest mètode, així que vaig haver de recórrer a un altre opció: les fàbriques que realitzen prototips de circuits impresos.

3.3.1. Selecció de la fàbrica de prototips

Normalment segueixo a diari diversos blogs sobre tecnologia i havia anat recopilant informació sobre empreses que creen circuits impresos en petites quantitats i l'experiència de diversos usuaris. Em va cridar l'atenció una empresa xinesa, la *Gold Phoenix PCB Co., Ltd.* Els resultats semblaven ser molt bons i els preus també eren molt ajustats.

Gold Phoenix Printed Circuit Board Co.,Ltd
金 凤 凰 印 刷 电 路 板 有 限 公 司

Il.lustració 3.6: Logotip de Goldphoenix

L'experiència d'un parell d'usuaris em va convèncer a seleccionar aquesta empresa. A més, les capacitats tecnològiques de la fàbrica s'ajustaven perfectament a les meves necessitats. A la Taula 1 es mostren les capacitats de fabricació per a circuits estàndard (com el meu) i fins a quin grau de complexitat poden oferir en altres tipus de treballs.

Característica	Circuit estàndard bàsic*	Capacitats reals de l'empresa
Àrees del panell	1000cm² o bé 650cm²	Consultar
Capes	2 o 4	24
Material	FR4 (TG130/170)	FR4(TG130/170)
Tests de qualitat	Elèctric	Elèctric, Rajos X, Microscopi 20X
Amplada i espai de pistes	7mil	3mil
Tamany del forat	15mil	4mil
Acabats de la superfície	HASL, OSP, Immersió en Plata/Or/Estany, Altres acabats en Or i Plata especials.	
Altres característiques	- Compliment del RoHS disponible - Serigrafia (Diversos colors) - Màscara de soldatge (Diversos colors)	

Taula 1: Capacitats tecnològiques de Golg Phoenix PCB Co., Ltd.

**Pagant tarifes extres es pot aconseguir millor precisió i reduir aquests límits.*

Amb aquesta taula queda demostrada la qualitat amb què són capaces de produir empreses xineses i com evoluciona el gegant asiàtic, no només en quantitat de producció sinó també en la seva millora tecnològica i la qualitat.

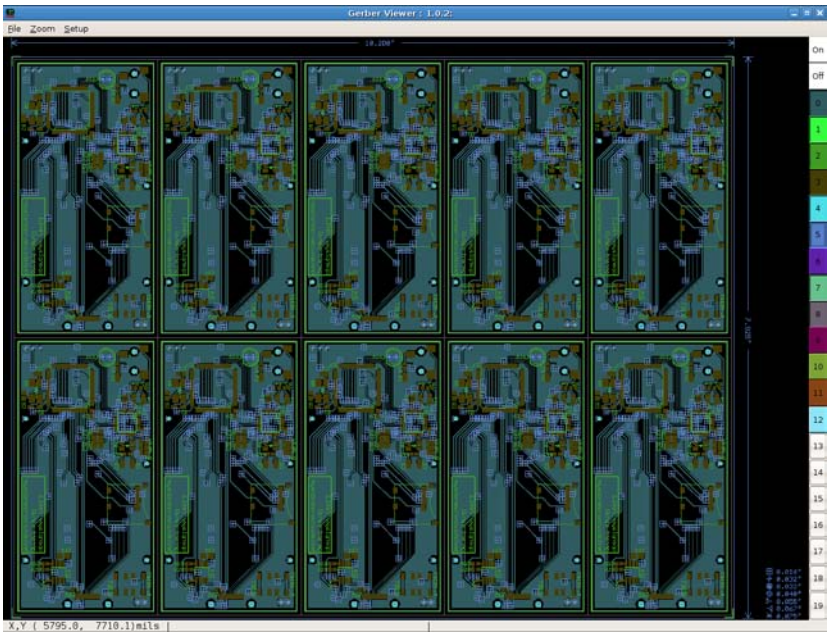
3.3.2. Generació dels fitxers estandarditzats Excellon i Gerber

Ja que existeixen molts programes de disseny de circuits, en el moment d'enviar a una industria el nostre circuit perquè sigui fabricat cal transformar el fitxer del programa de treball a uns models estàndards. El programa *Eagle* incorpora aquesta funció i coneixent les capes de la nostra placa i el que es

vol incloure en el procés de fabricació (capa superior, inferior, serigrafia...) es generaran els fitxers necessaris. Uns dels formats acceptats per la majoria d'empreses són els *Excellon* i *Gerber*. Els fitxers *Excellon* contenen informació corresponent als forats que conté la placa: vies, forats de col·locació... i els fitxers *Gerber* contenen la placa en sí: zona de coure, zones d'impressió de la màscara de soldar, serigrafia, etc.

3.3.3. Panelització i enviament

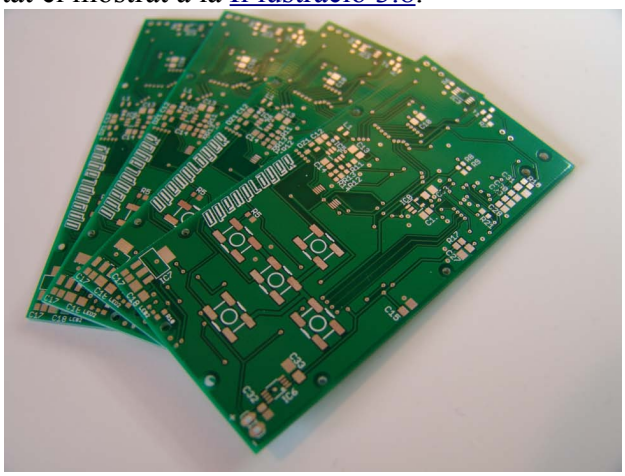
Els circuits impresos no es fabriquen individualment, sinó que se'n fa una unió en un panell de superfície determinada[8]. Aquest procés s'anomena *panelització* i es realitza unint el conjunt de fitxers *Gerber/Excellon* generats anteriorment que només són d'una placa individual. Per fer-ho, he utilitzat un programa anomenat *gerbmerge*, el qual mitjançant una sèrie d'indicacions i les mides del panell màximes genera automàticament el panell juntament amb informació extra pel fabricant[9].



Il.lustració 3.7: Panelització de les plaques de l'openplayer

3.4. Resultat

Un cop enviades al fabricant, les plaques es reben en un període de menys de 10 dies. En el cas de l'openplayer, el resultat ha estat el mostrat a la Il·lustració 3.8.



Il.lustració 3.8: Les plaques de l'openplayer

8 Vegeu "Àrees del Panell", Taula 1
9 Podeu accedir a els fitxers Gerber/Excellon (panelitzats i no panelitzats) a l'Annex 5, apartat 2.2

4. Ensamblat dels components

Un cop arriben les plaques, és necessari col·locar i soldar tots els components sobre la placa. És un procés molt laboriós, especialment en aquest cas on la majoria de components són de tamany molt petit i requereixen soldar amb precisió. És clar que també es podria haver realitzat un ensamblat professional de fàbrica (Gold Phoenix també ho fa) tot i que no m'ho vaig plantejar bàsicament pels costos i a més, els primers prototips són propensos a contenir errors com es veurà en el capítol següent. Per tant és absurd invertir diners en l'ensamblat del primer prototip si aquest es pot soldar a mà i pot ser la versió no definitiva

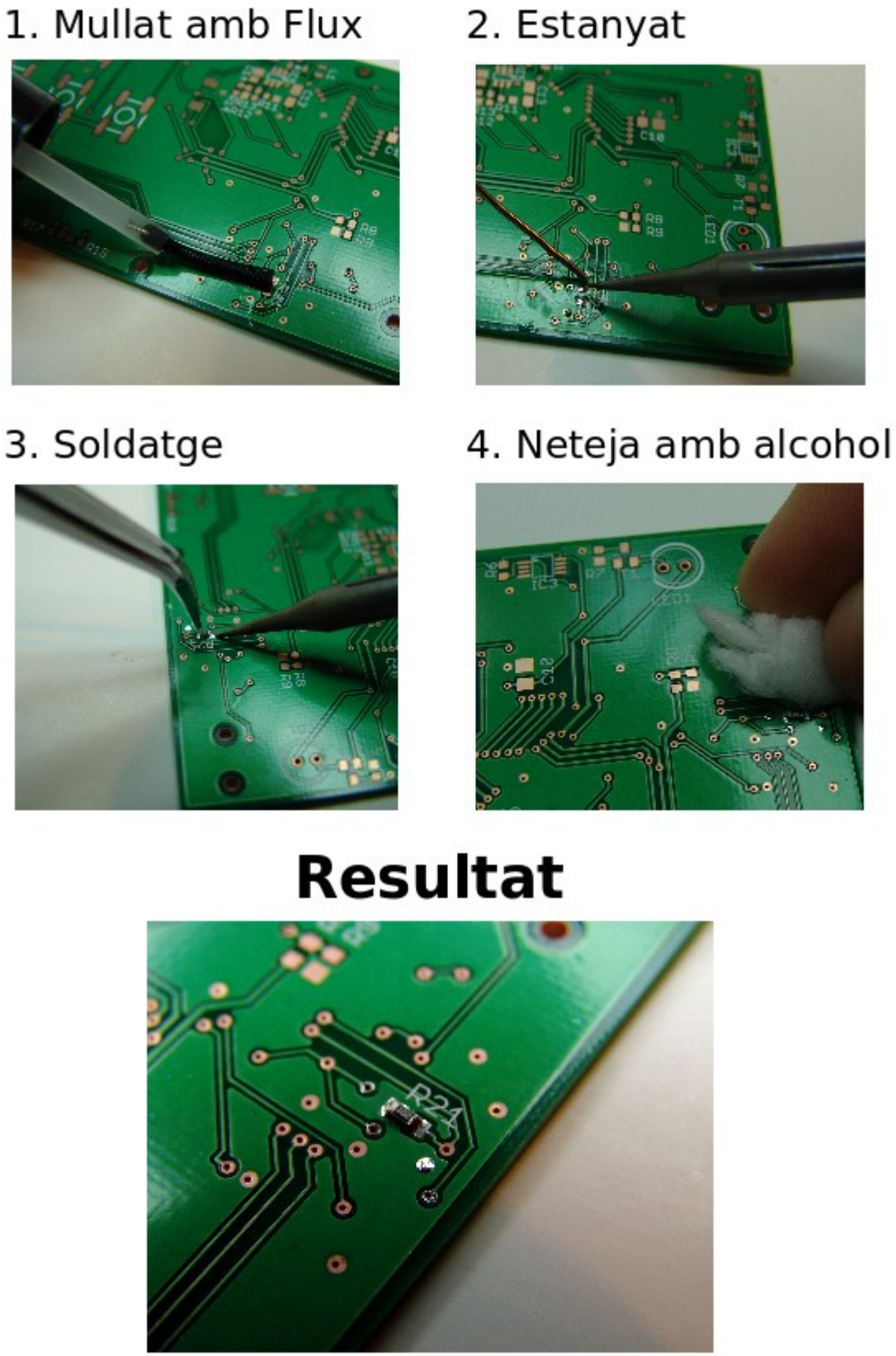
4.1. Eines de soldatge

Per soldar l'openplayer, he utilitzat el conjunt d'eines que es descriuen i es mostren a continuació:

1. **Soldador:** Eina essencial, utilitzada per fondre l'estany sobre les pistes i per tant soldar els components.
2. **Estany:** Metall que té un punt de fusió baix utilitzat en el soldatge de components electrònics.
3. **Flux:** Líquid posat sobre la superfície que s'ha de soldar per millorar l'adherència i facilitat de soldatge de l'estany.
4. **Pinces de soldatge:** Pinces especials per soldar, utilitzades per subjectar els components abans de ser units.
5. **Lupa:** Utilitzada com a mitjà ampliador del circuit a l'hora de soldar, ja que algunes pistes són de tamany molt petit i és de gran ajuda poder treballar amb una visió augmentada.
6. **Fil Wrapper:** Fil de secció molt prima utilitzat per unir punts del circuit erronis o pistes que no s'han posat.
7. **Tira desoldadora:** En el cas que algun component hagi d'ésser extret del circuit un cop ja soldat, s'utilitza aquesta tira per retirar l'estany i tornar a deixar el lloc lliure.

Il.lustració 4.1: Eines de soldatge

4.2. Procés de soldatge



Il.lustració 4.2: Procés de soldatge

Ja que una explicació teòrica del procés de soldatge seria poc útil i difícil d'imaginar, vaig decidir il·lustrar el procés amb fotografies. A la Il·lustració 4.2 es mostra aquest procés on també es pot identificar

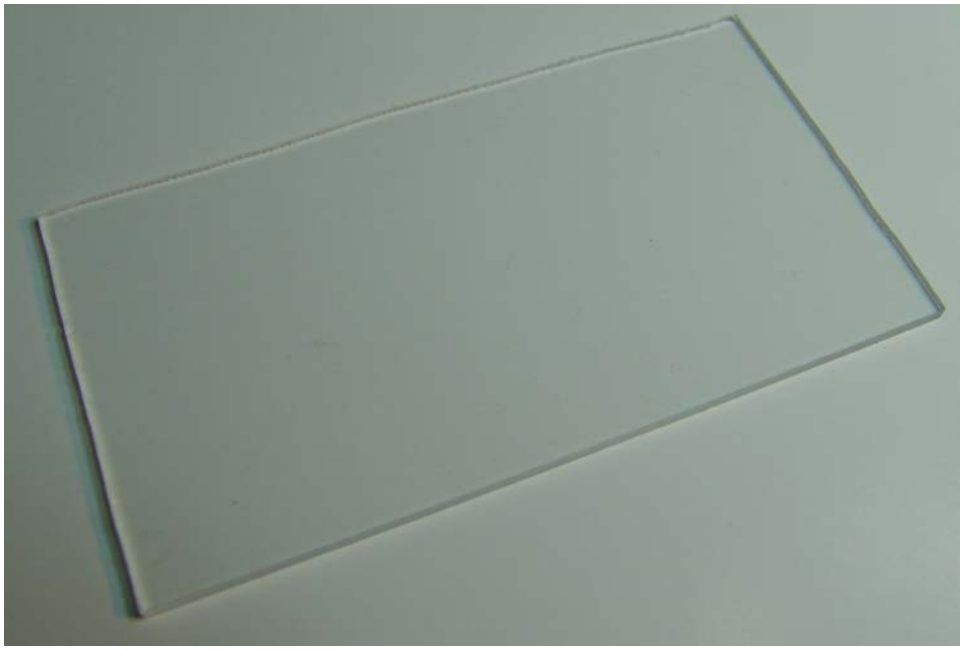
alguns dels components descrits en el punt anterior.

5. La carcassa

La carcassa, l'objecte protector de tot el circuit i components, ha estat l'últim pas de l'openplayer. La veritat és que feia temps que tenia una idea, però no en sabia ni l'estructura ni tampoc com fer-la. M'hauria agradat disposar de més temps i materials per poder-la crear i dissenyar amb més cura com també disposar d'una talladora làser, on hauria pogut obtenir uns tallats perfectes, però el pressupost m'ha impedit accedir-hi.

5.1. Material seleccionat

El material sempre va ésser molt clar: el metacrilat. Vaig triar aquest material perquè era de color transparent, i per tant permetria observar l'interior del reproductor (circuits, bateria, etc.) que de fet corresponen al treball de veritat. És un material poc manejable i difícil de treballar si no es disposen de les eines necessàries com és el meu cas, però amb paciència es poden aconseguir bons resultats.



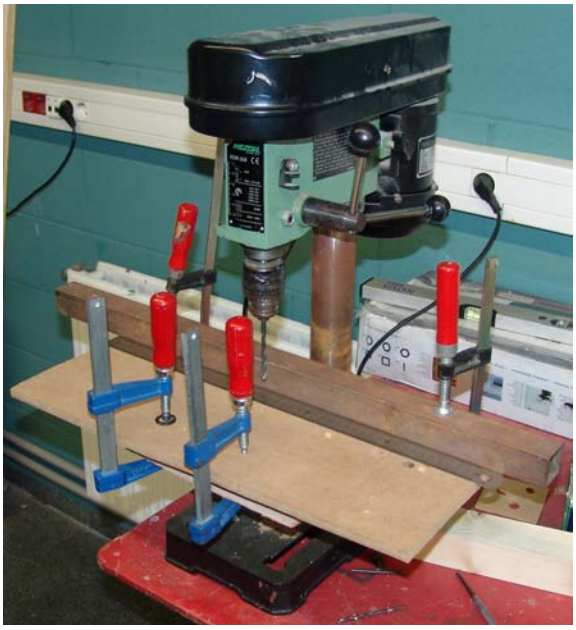
Il.lustració 5.1: Tall de metacrilat

5.2. Procés de construcció

El procés de construcció de la carcassa ha estat més llarg del que em pensava, ja que per falta d'eines adequades ha calgut repetir diversos cops el tallat d'algunes peces. En els següents punts s'explicarà un resum d'aquest procés de construcció.

5.2.1. Estructura

L'estructura (superfície i laterals) sempre havia pensat que seria la part més ràpida ja que simplement era tallar rectangles i fer-hi els forats corresponents, tot i que al final ha estat el procés més llarg. Per fer-ho he utilitzat les serres mecàniques i trepants disponibles al taller de l'institut, les quals han facilitat molt el procés de construcció. En molts casos he hagut de repetir el procés perquè el metacrilat es trenca amb facilitat si no es tracta suaument. No he realitzat plànols amb ordinador de la carcassa per falta de temps així que com es pot veure ha estat feta amb pocs apunts sobre algun paper. A les fotografies que es mostren a continuació es pot observar el material utilitzat i el procés de trepanat.



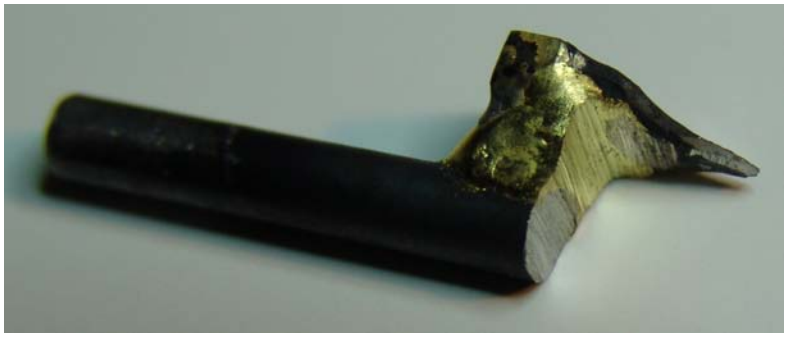
Il.lustració 5.2: Trepant utilitzat per fer forats

Il.lustració 5.3: Serra mecànica utilitzada per tallar

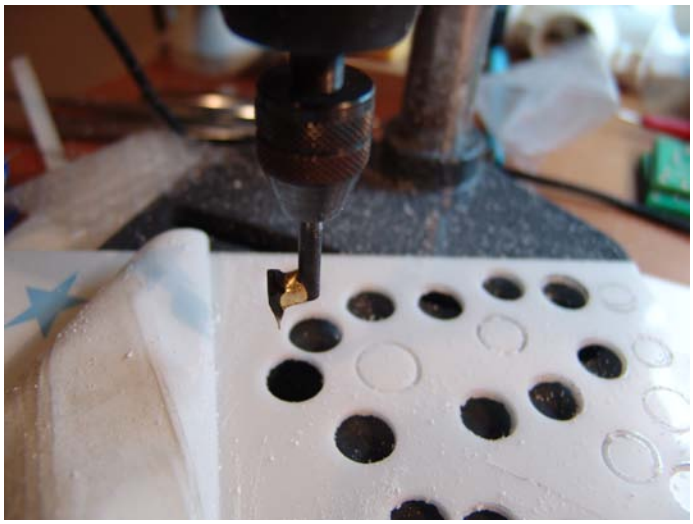
Il.lustració 5.4: Procés de trepanat

5.2.2. Tecles

A diferència de la carcassa, la creació de les tecles ha estat un procés molt ràpid. Va ser difícil imaginar com faria circumferències de 12mm de diàmetre. Vaig cercar en diversos llocs alguna eina que fos capaç de fer-ho, però no vaig trobar res. Finalment, vaig decidir demanar ajuda al meu pare i varem recórrer a una altra solució casolana. Varem crear una broca en forma de compàs que creés circumferències d'aquell diàmetre. Curiosament la broca va sortir correcte en el primer intent. A les fotos següents es mostra aquesta broca i el resultat de la creació de tecles.



Il.lustració 5.5: Broca per crear circumferències



Il.lustració 5.6: Trepanat de les tecles

Il.lustració 5.7: Tecles muntades

5.2.3. Unió

Per unir les diverses peces de metacrilat, va caldre comprar un fixador especial ja que no es pot utilitzar qualsevol cola. El fixador que vaig comprar és l'*Acrifix 192*, especial per fer unions entre peces de metacrilat. És un material irritant i s'han de prendre precaucions en manejar-lo: portar guants, mascareta i ulleres protectores. A les següents fotografies es pot observar el producte i el procés d'unió.

Il.lustració 5.8: Acrifix 192

Il.lustració 5.9: Unió del metacrilat mitjançant les mesures de protecció adequades

Capítol 5

Resultats i conclusions

1. Les correccions a la placa

Quan vaig fer el primer model de la placa, no vaig tenir en compte que en el nou descodificador és necessari separar els voltatges de les diferents unitats ja que no totes poden treballar als 3.3V del circuit general. La versió anterior (VS1011) no tenia aquest problema. Aquest fet va comportar que la primera versió fos inestable reproduint música: alguns cops no sonava, d'altres es penjava... La veritat és que em va decebre molt, i a més, al principi no sabia l'origen de l'error. Vaig provar d'afegir més condensadors per filtrar rascant la placa però l'error persistia. Va ser gràcies a en Qibo Zhang qui em va notificar de la equivocació en el circuit. Vaig decidir mirar-me més a fons la fulla d'especificacions del descodificador, i aquest cop vaig seguir-la al preu de la lletra per crear la revisió 1.1. A més, aquesta nova versió inclou suport pel recent descodificador VS1053, una versió que afegeix suport OGG llençada per l'empresa VLSI el passat desembre. La revisió 1.1 també corregeix un error important que impedia el funcionament del comandament de TV: el transistor que el compon tenia l'emisor i el colector invertits. Tot i això aquest problema es pot solucionar fàcilment amb un petit fil a la revisió 1.0. També s'hi han arreglat altres petits detalls, quedant en total la llista següent de correccions:

- Suport complet i estable pels descodificadors VS1033/VS1053
- Corregit el patillatge del transistor del comandament de TV
- Canvi del format de paquet del regulador principal, cosa que permet afegir el botó "E. RESET" (reset elèctric, para la corrent al circuit mentre es troba pulsat).
- Canvi del regulador de voltatge de la bateria per un amb menys caiguda de voltatge (*drop-out*).
- Optimització d'algunes pistes
- Inclusió del logotip original a la placa

M'hauria agradat fer algunes correccions més, com per exemple incloure la bateria amb connector i no soldada, que hauria sigut molt més còmode. Tot i així vaig haver de fer les modificacions amb menys de dos dies i no vaig tenir temps.

2. Costs de creació

Crec que és important aturar-se a pensar en el cost total de l'openplayer. He hagut de comprar molt de material que no tenia, com per exemple el programador/debugger JTAGICE mkII. Moltes d'aquestes eines són cares, però que probablement podré utilitzar en futurs projectes. També he hagut de comprar molts components que no s'utilitzen a l'openplayer com per exemple diversos models de tecles, que és difícil elegir-les correctament a través de catàleg sense poder provar-ne el seu tacte. El fet d'haver de crear la revisió 1.1 també ha implicat un cost extra, però per mi era essencial entregar una versió estable fos com fos. Finalment crec que he estat afortunat de poder utilitzar programari lliure en gairebé la totalitat del projecte (compilador, sist. operatiu...) cosa que ha permès també la reducció dels costs de creació. Pel que fa purament als components electrònics, podeu veure'n els seus costs accedint a l'Annex 3.

3. Conclusions

L'openplayer ha estat el projecte més seriós i rigorós que he realitzat fins ara. Ha estat un projecte realment fascinant i on he dedicat moltes hores, a vegades potser més de les recomanables. Crec que la meva concepció del món de la informàtica i l'electrònica digital ha canviat molt, se m'han obert moltes portes on poder seguir creant i investigant i he après moltes coses noves que em fan veure que cada cop sé menys del que pretenia anteriorment. També m'ha permès veure en detall com es construeixen els dispositius electrònics, els costs de desenvolupament que tenen i el nombre d'hores que requereixen. En el meu cas aquestes hores no eren un problema, però en el món laboral sí que ho són: costen uns diners i els terminis d'entrega són altament rigorosos. Després d'haver acabat puc constatar que les coses no solen sortir en el primer intent, i si ho fan cal examinar-ho bé perquè podem trobar-nos amb sorpreses.
La utilització majoritària de programari lliure m'ha permès veure que el seu nivell de qualitat és molt alt i

que segueix progressant satisfactòriament gràcies a milers de persones i empreses que col·laboren dia a dia per millorar-lo. Crec que el programari lliure és una alternativa totalment viable per a molts projectes, i a més, evitar el programari privat és sempre una pràctica més sana per a qualsevol usuari o empresa. El fet de conèixer a en Qibo Zhang i de fabricar les plaques a la Xina també m'ha fet adonar del potencial dels països asiàtics, on sembla que tecnològicament (no ambientalment) avancen molt ràpidament, a diferència d'aquí.

Finalment, aquest treball m'ha servit per confirmar les meves sospites i els comentaris en general sobre l'estat del nostre país tecnològicament: sota mínims. El desenvolupament i la recerca semblen ser temes ignorats per la nostra classe política. El més trist de tot això és que aquelles persones que desitgen estudiar i progressar es vegin obligades a marxar del país per poder continuar les seves tasques d'investigació. Crec que seria necessari un replantejament del nostre sistema educatiu (no només en els centres secundaris i primaris, sinó també en universitats) i reformar-lo íntegrament. Trobo totalment normal que moltes empreses no apostin en el nostre país com a lloc on implantar centres d'investigació, jo tampoc ho faria.

Bibliografia

Llibres

SCHILDT, Herbert. *Programación en C, Cuarta edición*. Madrid: McGraw Hill, 2001

SCHERZ, Paul. *Practical Electronics for Inventors, Second Edition*. Estats Units: McGraw Hill, 2007

Documents i publicacions

ATMEL, "AVR035: Efficient C Coding for AVR"

ATMEL, "AVR130: Setup and Use the AVR Timers"

AVR-LIBC, "avr-libc Reference Manual 1.4.6"

BANNACK, Angelo i BRUNO, Giordano. "FAT16/32 Driver for ATMEL AVR".

Cadsoft, "Eagle Manual 3.55 and later"

CAMERA, Dean. "GCC and the PROGMEM Attribute"

CAMERA, Dean. "Using the EEPROM memory in AVR-GCC"

CARTER, Bruce; Texas Instruments Incorporated. "The PCB is a component of op amp design"

EVANS, Allan; Dept. of Electrical and Computer Engineering. "Fat16 Interface for MSP430"

FOUST, F; Dept. of Electrical and Computer Engineering, "Secure Digital Card Interface for the MSP430"

ID3, "ID3 tag version 2.3.0"

MAXIM, "APPLICATION NOTE 162: Interfacing the DS18X20/DS1822 1-Wire Temperature Sensor in a Microcontroller Environment "

O'FLYNN, Collin, "Using AVR-GDB and AvaRICE Together"

VLSI, "VS10XX Application notes"

WINDSZUS, Konrad. "MPEG Audio Frame Header"

Datasheets: Vegeu l'Annex 5, apartat 2.3

Pàgines web

http://atlc.sourceforge.net

http://electrons.psychogenic.com

http://instruct1.cit.cornell.edu/courses/ee476

http://lists.nongnu.org/mailman/listinfo/avr-gcc-list

http://www.atmel.com/avr

http://www.avrfreaks.net

http://www.avrrepository.com

http://www.codeproject.com

http://www.digitalspirit.org

http://www.e-armazem.com.br/

http://www.elm-chan.org

http://www.eng.hawaii.edu

http://www.fortunecity.com/skyscraper/windows/364/

http://www.fpga4fun.com

http://www.imagendv.com

http://www.instructables.com

http://www.makezine.com/blog

http://www.makingthings.com

http://www.mictronics.de

http://www.multiweb.cz/twoinches

http://www.nongnu.org/avr-libc

http://www.opencircuits.com

http://www.retroleum.co.uk

http://www.sbprojects.com/

http://www.thomaspfeifer.net

http://www.tuxgraphics.org

http://www.wikipedia.org

http://www.yampp.com

Alguns dels documents llistats en aquesta bibliografia es poden obtenir accedint a l'Annex 5, que correspon al contingut digital en un DVD.

Annexos

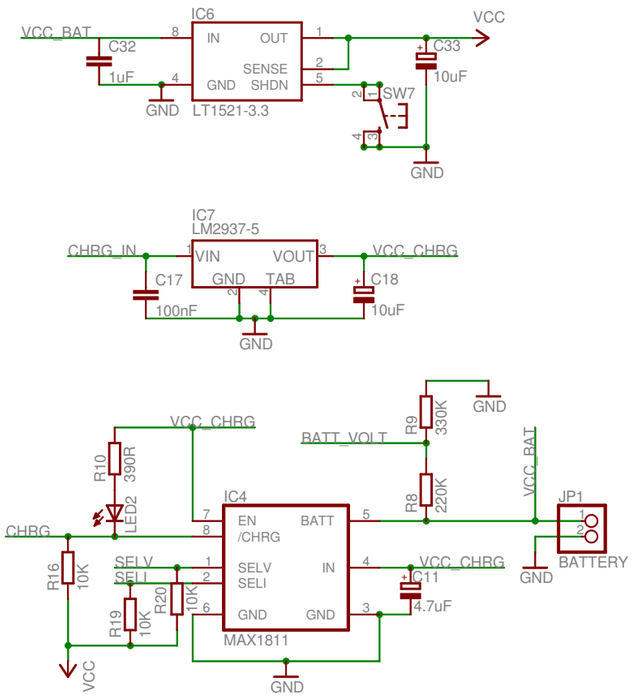
Annex 1

Esquemes i circuits

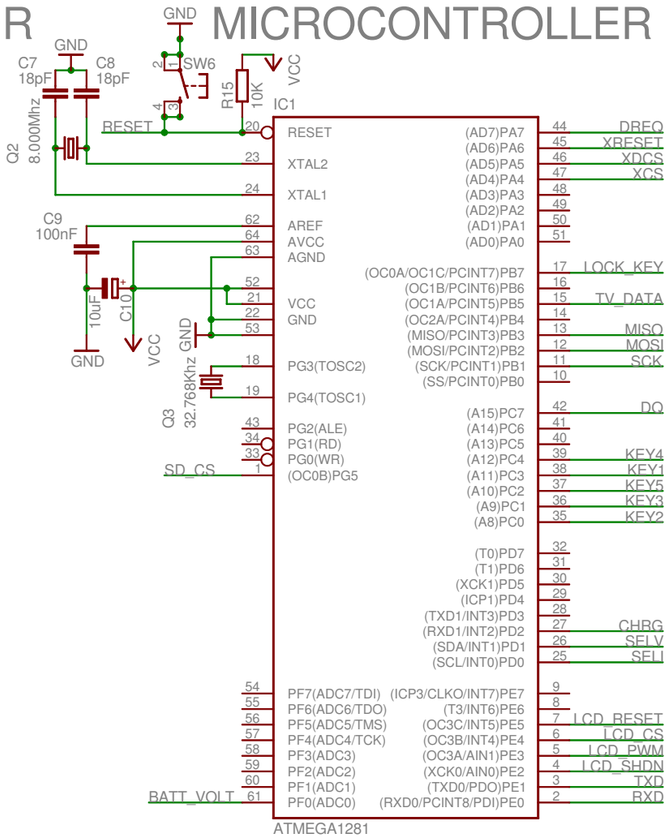
1. Informació

En aquest annex s'hi troba l'esquema elèctric de l'openplayer (només la revisió 1.1) juntament amb les imatges del circuit imprès (revisió 1.0 i 1.1). També he decidit incloure una versió original de les dues revisions de les plaques perquè el lector les pugui observar en major detall. Les plaques incloses es poden treure del sobre i tocar sense cap problema. Es recomana visualitzar els esquemes utilitzant les seves versions digitals mitjançant un ordinador, accessibles des de l'Annex 5 corresponent al DVD.

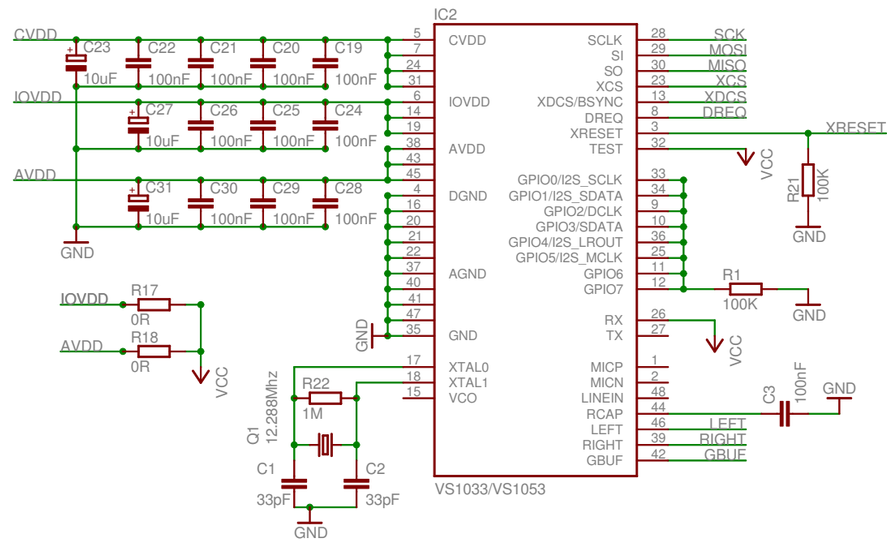
POWER SUPPLY & BATT. CHARGER

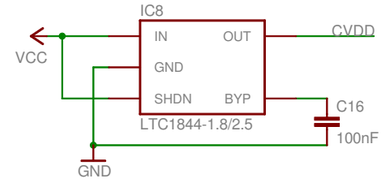


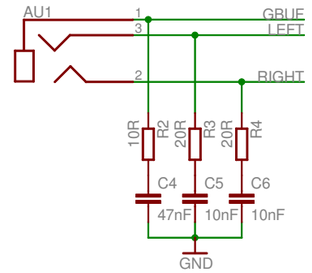
MICROCONTROLLER



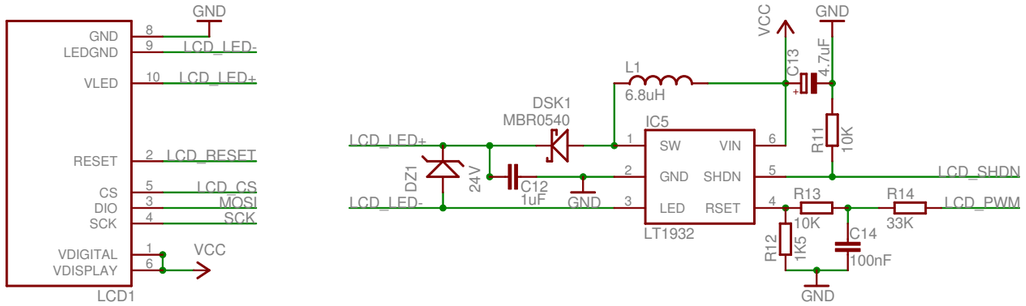
AUDIO DECODER



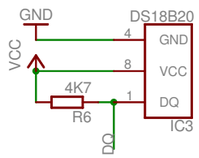




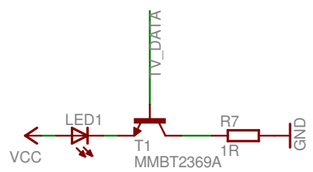
LCD & BACKLIGHT DRIVER



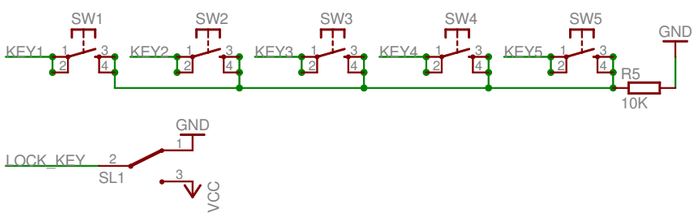
TEMP. SENSOR



TV REMOTE

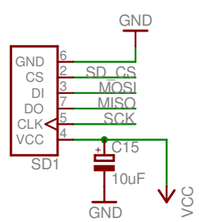


KEYBOARD & LOCK KEY



DOCK

MICROSD CARD

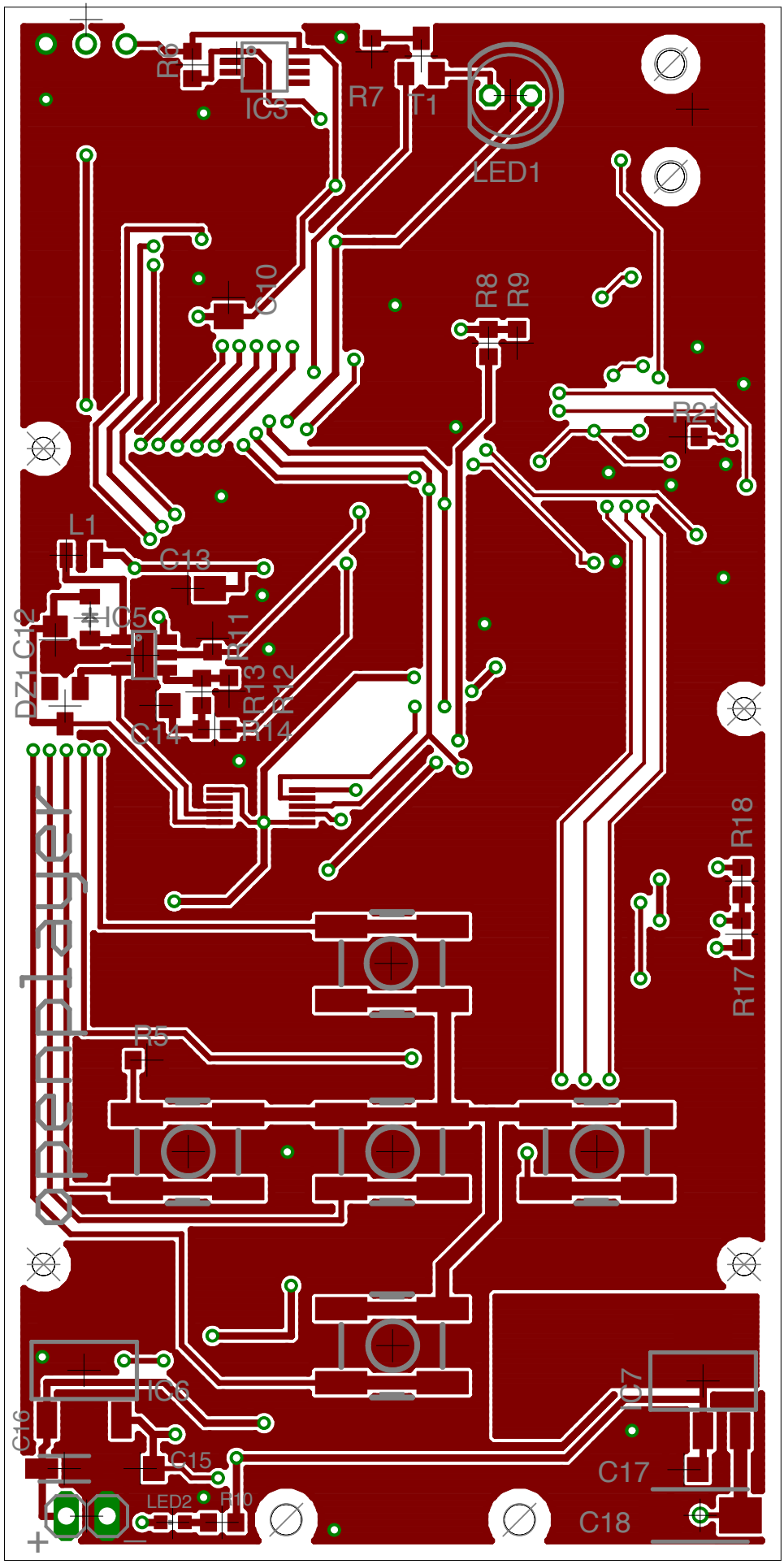


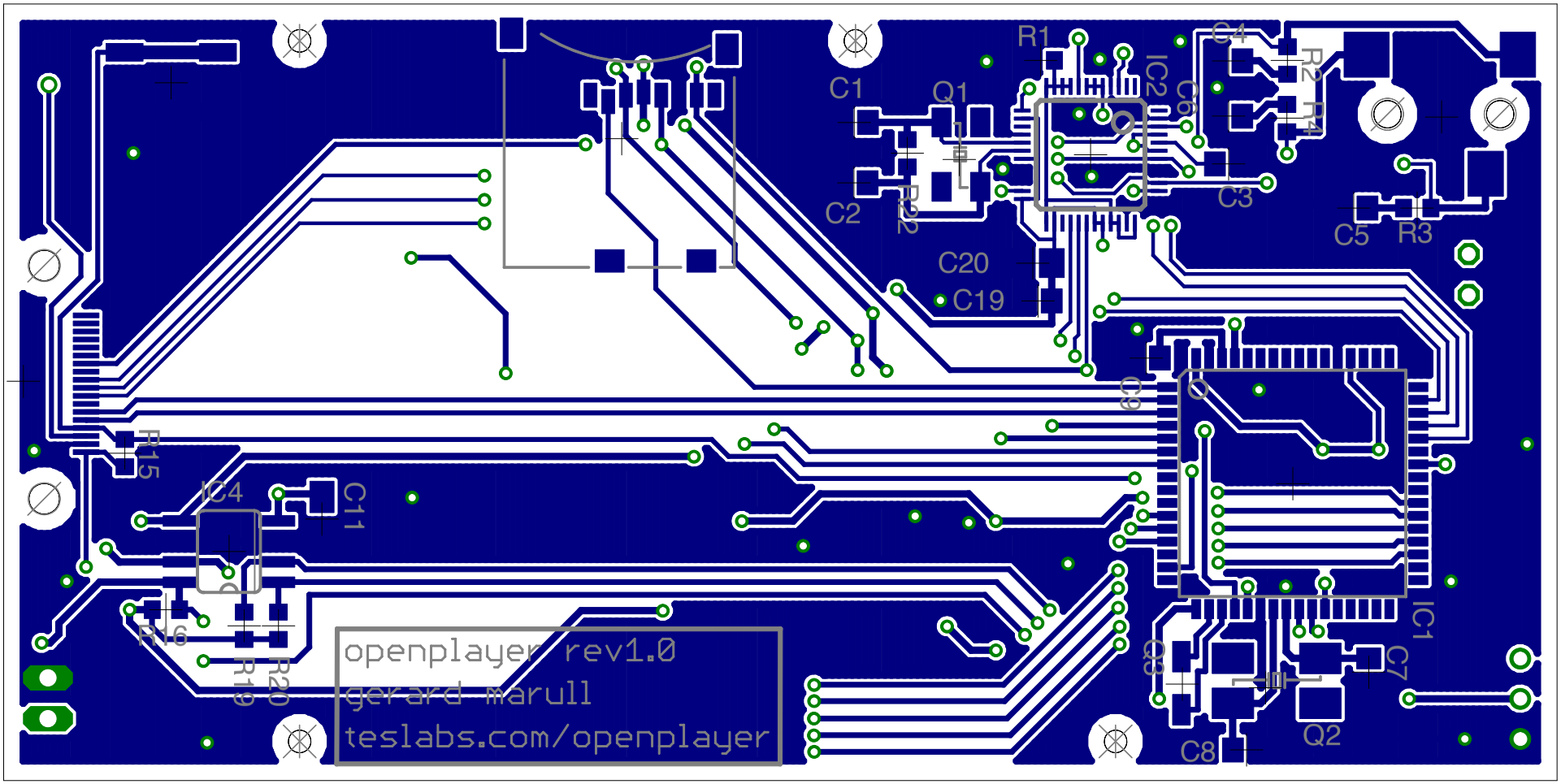
OPENPLAYER

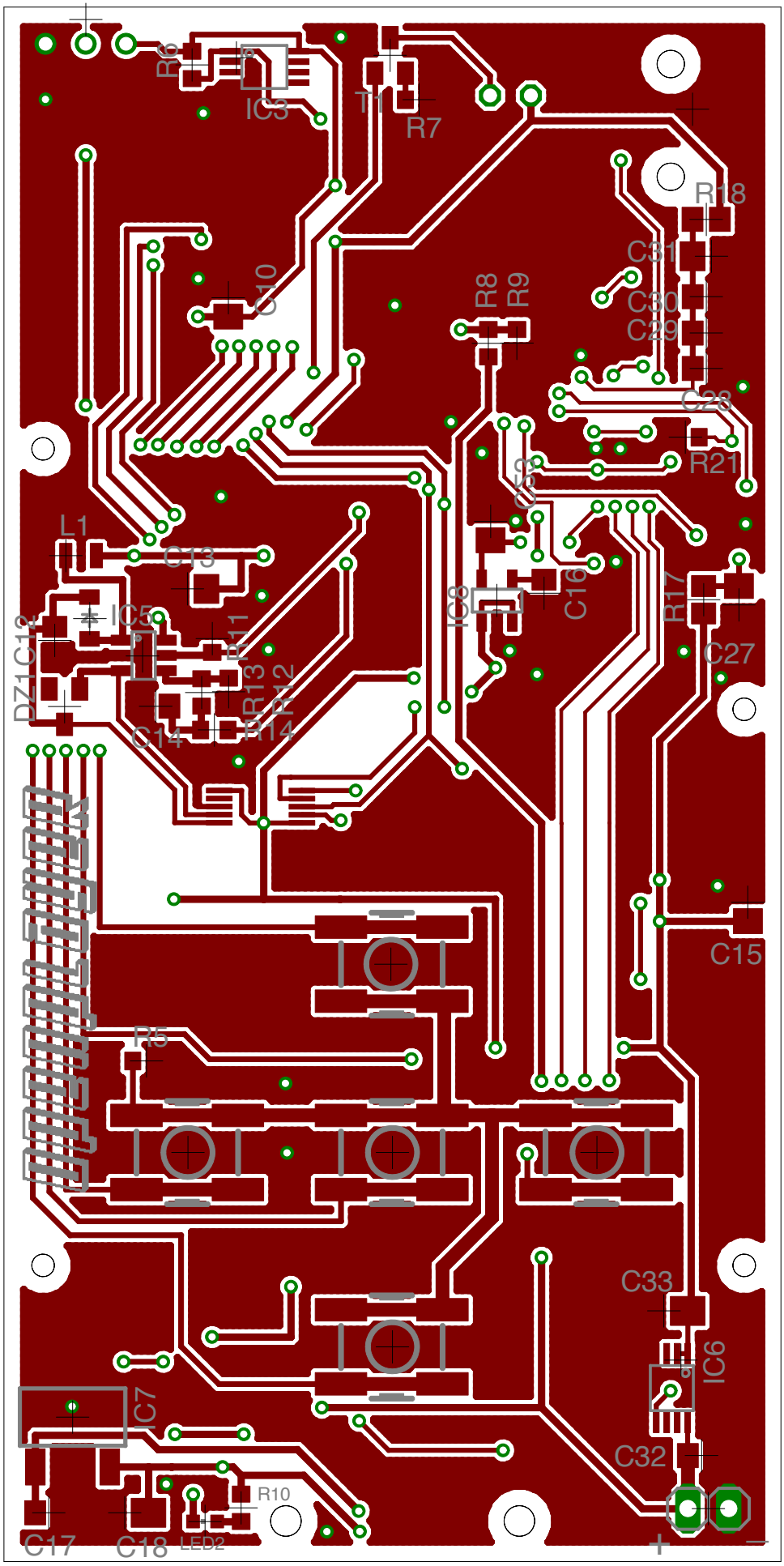
A music player for geeks

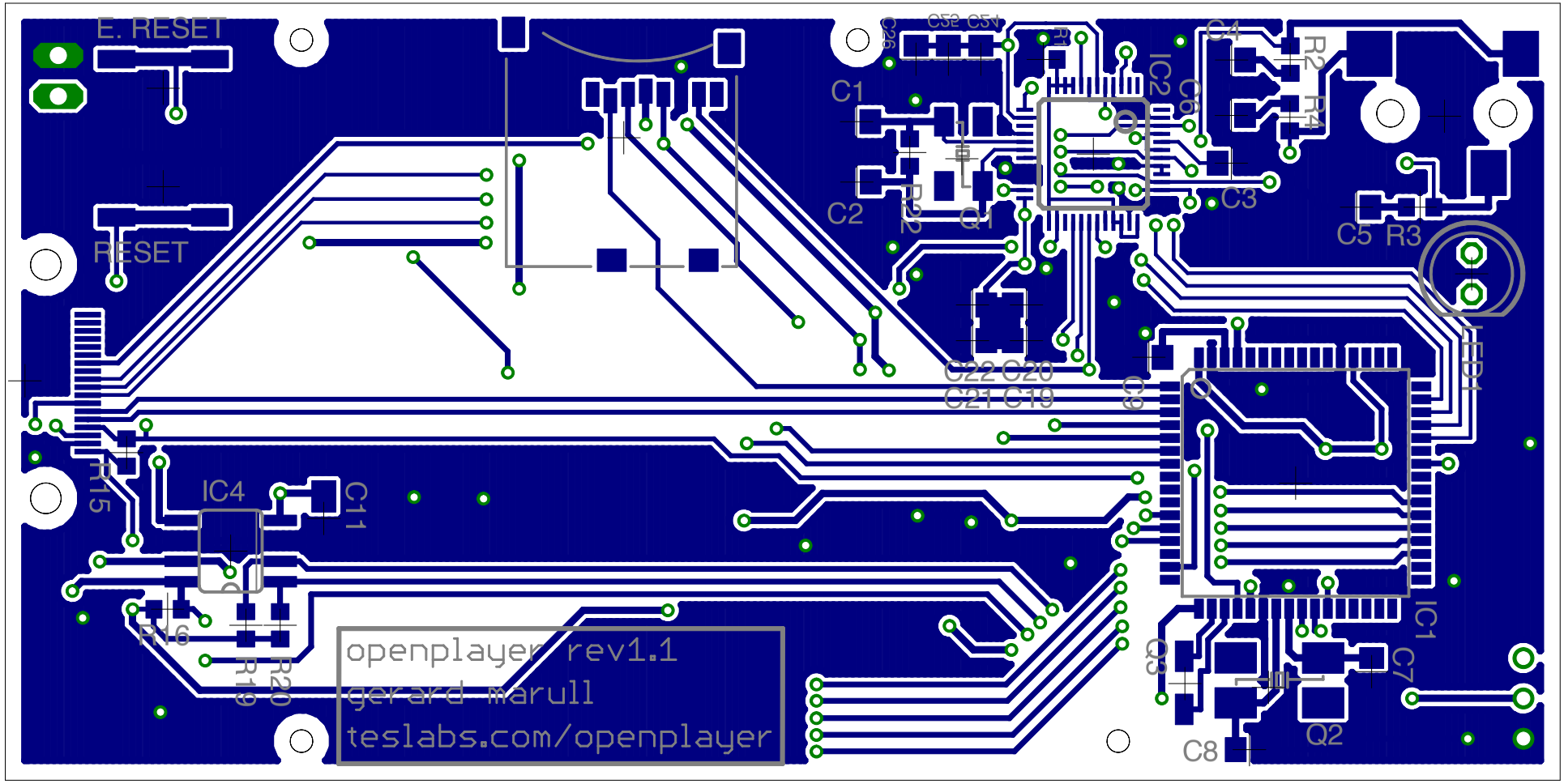
Board revision 1.1

Author	Gerard Marull Paretas
Project	openplayer
Date	1/09/2008 22:24:34
Sheet	1/1


R6
IC3
R7
T1
LED1
C10
R8
R9
R21
L1
C13
IC5
DZ1
C12
R11
R13
T2
R14
C14
R18
R17
R5
greenlayer
IC6
C16
C15
LED2
R10
C17
C18


openplayer rev1.0
gerard marull
teslabs.com/openplayer
C1
C2
C3
C4
C5
C6
C7
C8
C19
C20
R1
R2
R3
R4
R5
R6
R15
R16
R17
R18
Q1
Q2
Q3
IC1
IC2
IC4


R6
IC3
T1
R7
R18
C31
C30
C29
C28
R21
C10
R8
R9
C9
C16
IC8
R17
C27
L1
C13
IC5
C11
C12
DZ1
R13
T2
R12
R14
C14
C15
R5
C33
IC6
IC7
C32
R10
C17
C18
LED2
+
–


E. RESET
RESET
C1
C2
C3
C4
C5
C6
C7
C8
C11
C19
C20
C21
C22
C26
C31
C32
R1
R2
R3
R4
R12
R15
R16
R18
R20
IC1
IC2
IC4
Q1
Q2
Q3
Q8
LED1
openplayer rev1.1
gerard marull
teslabs.com/openplayer

Annex 2

Fabricació de circuits impresos

1. Introducció i materials

En aquest annex s'explicarà el procés de construcció de circuits impresos que he utilitzat en algunes plaques de l'openplayer gran. Aquest procés utilitza com a base una impresora làser i un paper especial anomenat *Glossy paper*. Aquest paper és de superfície satinada i això permet transferir-lo després amb una planxa sobre una placa de coure, que finalment serà revelada mitjançant components químics. A la Taula 1 s'hi troba la llista de material necessari i també on es pot comprar cada cosa.

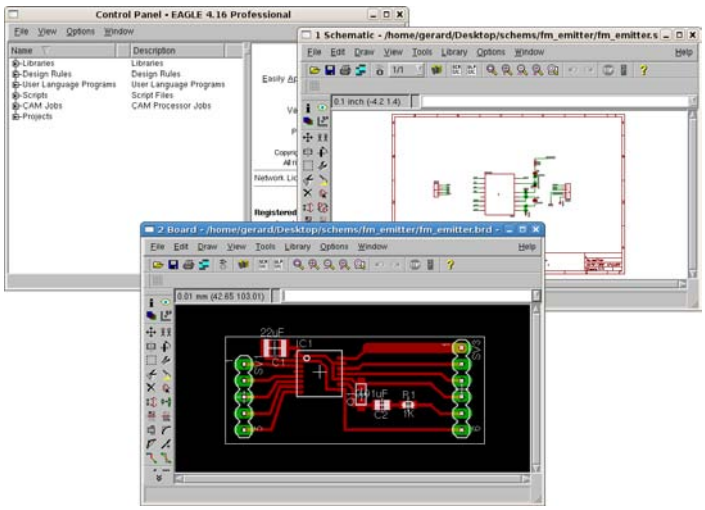
Disponibilitat	Material
■	Placa de coure verge (fibra de vidre o baquelita)
■ ■	Fulla A4 de paper satinat –*glossy paper*-
■	Impressora làser
■	Planxa domèstica
■ ■	Àcid Clorhídric (HCl) + Perborat sòdic ($NaBO_3$) o també Clorur Fèrric ($FeCl_3$)
■	Cubeta de plàstic
■	Màquina de foradar per a broques de 0.8/1mm amb suport
■	Esprai de laca/vernís protector/a especial per a circuits impresos
■	Dissolvent comú o Acetona

Taula 1: Material necessari per realitzar circuits impresos

■ - *Disponible en botigues d'electrònica*
■ - *Disponible en botigues d'informàtica*
■ - *Disponible en papereries*
■ - *Disponible en botigues de la llar / basars*
■ - *Disponible en botigues de components químics / drogueria*

2. Disseny

Abans de poder començar el procés de fabricació és necessari disposar d'un disseny fet amb ordinador. En aquest annex no s'explicarà el procés de disseny ja que no n'és l'objectiu. El programa utilitzat en aquest cas és l'*Eagle* tot i que se'n pot utilitzar qualsevol altre. Primerament es crea l'esquema elèctric i llavors aquest es transforma en un circuit imprès que llavors podrem imprimir. Mitjançant aquest procés es recomana no utilitzar pistes massa estretes ja que en alguns casos no queden ben revelades.



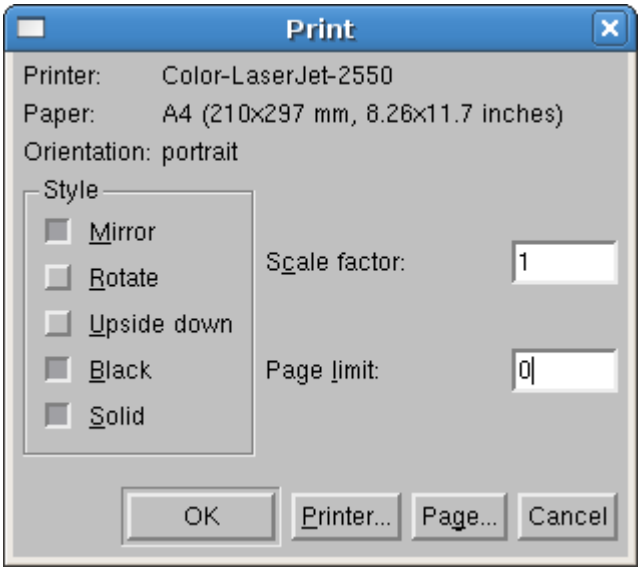
Il.lustració 2.1: Eagle

3. Impressió i transferència

Un cop acabat el disseny es procedirà a imprimir i transferir el circuit a la placa de coure. No es pot utilitzar un paper qualsevol, sinó que ha de ser un paper amb una superfície satinada. Podem trobar-lo anomenat com a *glossy paper*. Mitjançant una impressora làser s'imprimirà sobre el paper satinat el circuit, tot i que caldrà tenir en compte un paràmetre de la impressora: la seva resolució. Si el disseny no inclou connexions inferiors al mil·límetre, no serà cap problema, ja que la precisió en aquests casos no és tant important. Però en casos on aquestes separacions si que són de menys del mil·límetre es recomana la utilització d'una resolució com a mínim de 1200ppp ja que sinó algunes pistes poden quedar ajuntades o bé no revelar-se. La majoria d'impressores són capaces d'imprimir a aquesta resolució, però si existeix el dubte, caldrà informar-se en el full tècnic de la impressora.

3.1. Impressió mitjançant l'Eagle

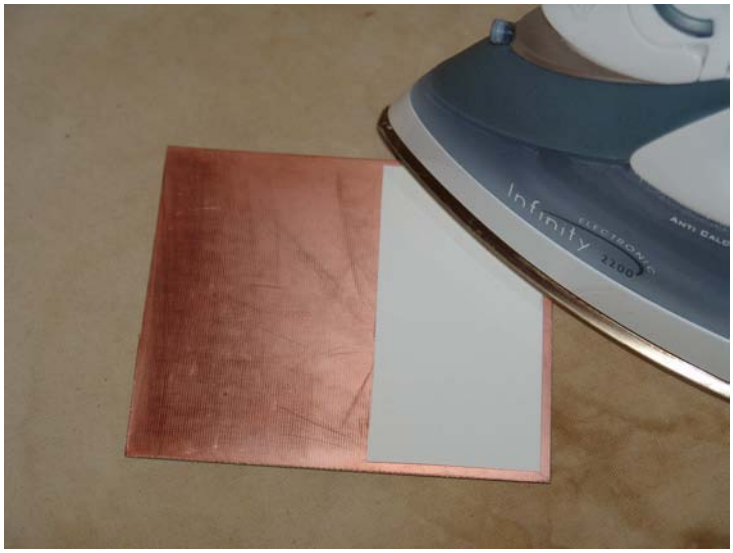
Abans de poder imprimir s'hauran de fer invisibles totes aquelles capes que no han d'aparèixer en el procés d'impressió. Es pot fer a través del botó  i seleccionar quines es volen mostrar o no. Un cop fet es podrà accedir al menú d'impressió localitzat a *File / Print*. En aquest menú s'hauran de seleccionar les següents opcions: *Mirror, Black i Solid;* tal com es mostra a la Il·lustració 3.1.



Il.lustració 3.1: Menú d'impressió

3.2. Transferència a la placa de coure

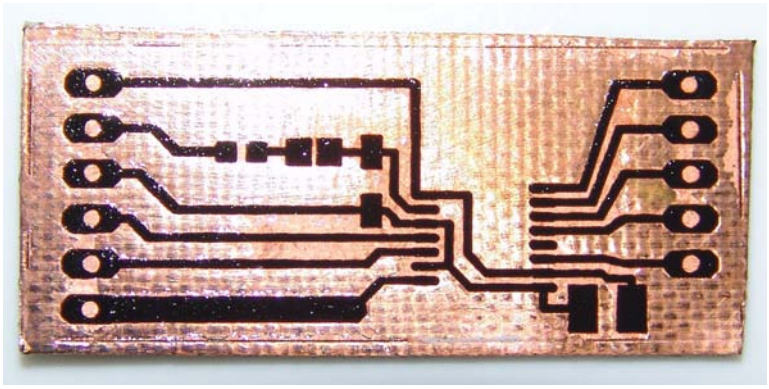
Per transferir el circuit imprimit sobre el paper satinat a la placa de coure s'utilitzarà una planxa domèstica ja que es necessita una superfície a alta temperatura. Primerament s'haurà de comprovar que la superfície del coure estigui en bones condicions i sense oxidació, en cas contrari s'haurà de netejar amb llana d'acer. Un cop comprovat, caldrà centrar bé el paper abans de començar a planxar. En casos de plaques de doble capa, aquest és un pas que s'ha de realitzar amb estricte precisió, ja que després alguns forats podrien no coincidir. Seguidament, s'aplicarà la planxa sobre el paper durant aproximadament 5 minuts (Il·lustració 3.2) exercint una petita pressió perquè quedi ben enganxat. Un cop acabat es podrà comprovar com el paper ha quedat totalment adherit a la placa de coure.



Il.lustració 3.2: Procés de planxat

3.3. Extracció del paper

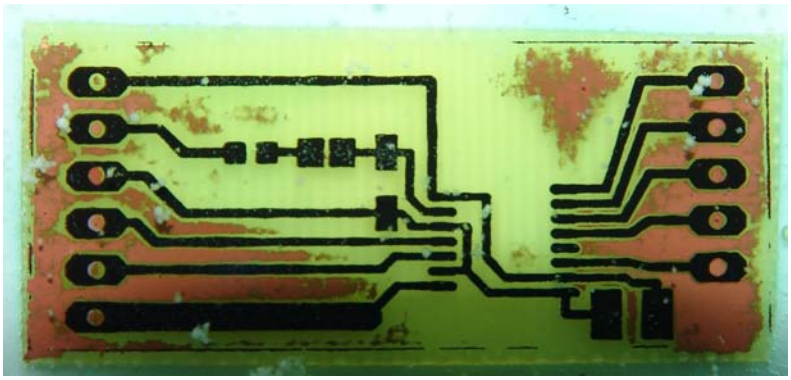
Per treure el paper que hem transferit però conservant el circuit a la placa només caldrà posar-lo en remull en una cubeta junt amb aigua (es recomana aigua temperada). Passats 5 minuts, temps perquè el paper hagi absorbit suficient aigua i sigui fàcil de treure es podrà arrancar de totes les zones. En cas de quedar petites restes de paper, es pot utilitzar un raspall de dents per treure-les. El resultat es mostra a la Il·lustració 3.3.

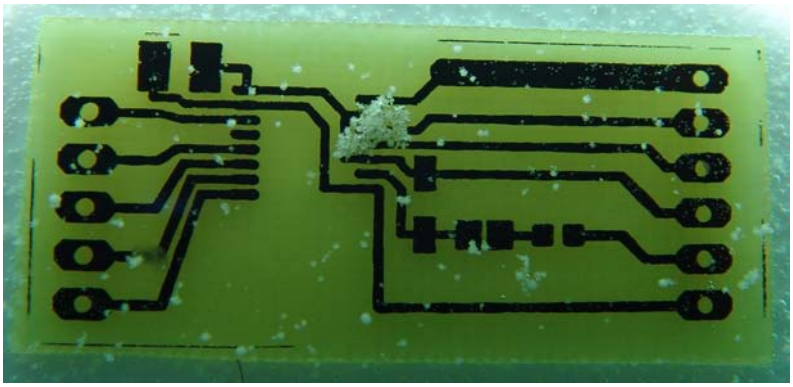


Il.lustració 3.3: Placa amb el paper netejat

4. Revelat

El procés de revelat consisteix en treure tot aquell coure innecessari i deixar només aquell que correspon a les pistes del circuit. Per realitzar aquest procés s'utilitzaran components químics que reaccionin amb el coure. Es pot utilitzar la combinació d'àcid clorhídric (coneguda també com a salfumant) i perborat sòdic o també clorur fèrric. La primera opció és la més ràpida ja que la reacció tarda pocs minuts. La segona és lenta i és convenient escalfar-la per agilitzar el procés. Un cop s'hagi seleccionat quin component químic es vol fer servir, s'abocarà aquest en una cubeta de plàstic juntament amb la placa. Un cop dins es podrà observar com comença la reacció química, la qual pot durar entre 5 i 10 minuts. És convenient moure la cubeta durant aquest procés. A les següents il·lustracions es pot observar un exemple d'aquest procés utilitzant la combinació d'àcid clorhídric i perborat sòdic:





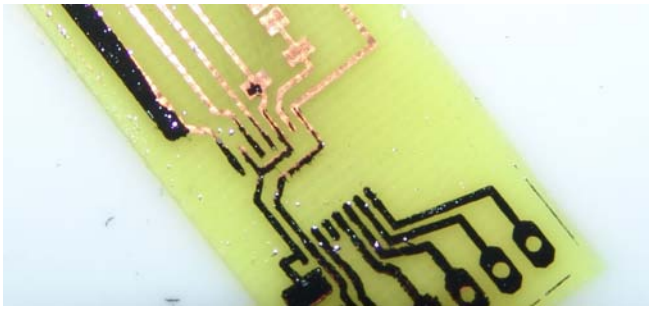
Il.lustració 4.1: Procés de revelat

5. Acabat

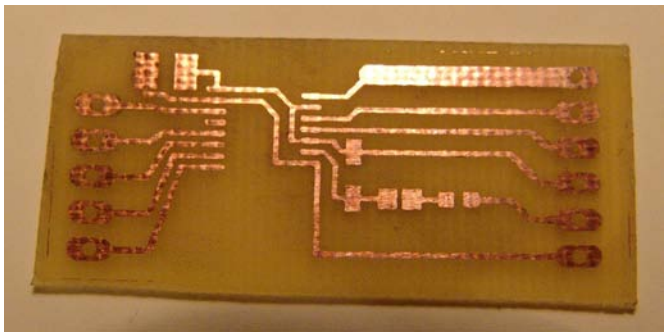
Ara tan sols queda treure la tinta de tòner que resideix sobre les pistes de coure, fer els forats en cas de ser necessari i protegir la placa contra possibles oxidacions. En els següents punts s'explicarà en detall cadascun d'aquests passos.

5.1. Neteja

Per netejar la tinta de tòner restant a la placa es pot utilitzar acetona o bé dissolvent comú. Només cal abocar-ho a sobre, esperar uns minuts perquè la tinta es desenganxi de la superfície i treure les restes mitjançant paper o cotó.



Il.lustració 5.1: Procés de neteja



Il.lustració 5.2: Placa totalment netejada

5.2. Foradat

En la majoria de plaques caldrà fer algun forat per col·locar components electrònics, pins de connexió, etc. Es necessitarà una màquina de foradar de tamany petit col·locada en un suport (també anomenat trepant, Il·lustració 5.3) per poder fer aquesta feina. Les broques necessàries poden ser de 0.8/1mm depenent del disseny de la placa.

Il.lustració 5.3: Trepant

5.3. Protecció

Finalment, es recomana aplicar una capa de laca/vernís en esprai especial per a circuits impresos per tal de protegir el coure de possibles oxidacions. Aquest procés cal fer-lo un cop s'han soldat tots els components, ja que un cop aplicat el vernís no es podrà tornar a soldar la placa.

Il.lustració 5.4:
Vernís protector

Annex 3

Costs de producció

1. Informació

És una mica difícil quantificar en realitat el que m'ha costat fer l'openplayer. En molts casos s'havien de comprar diversos components que fessin una mateixa funció per tal de veure quin era el millor. En altres casos alguns components no van servir, es van fer malbé o amb la placa ha hagut estar fabricada dos cops per corregir errors de la primera versió com també per afegir-hi noves funcionalitats. Tota aquesta llista de coses és coneguda com el cost de desenvolupament, i la veritat és que ha estat força alt, tant que m'he gastat tots els estalvis que tenia. En comptes d'incloure aquest cost he decidit incloure el cost del material de l'openplayer, que seria el cost real per un usuari si agafés aquest reproductor i decidís crear-se'n un. Ja que en electrònica els preus de producció varien molt en funció de la quantitat de producció, vaig decidir calcular el cost de producció unitari en el cas de fer-ne 10 unitats o bé 100 unitats. Es podrà observar que en el segon cas és més baix, i en cas de fer-ne més, encara ho seria més. També cal dir que s'informa del proveïdor corresponent de cada material per si alguna persona està interessada en obtenir algun dels components.

Components

Comp.	Qt.	Component	Format	Referència del proveïdor	Preu total [1]	Preu total [2]	Proveïdor
Circuits integrats							
IC1	1	Atmega1281	TQFP64	ATMEGA1281V-8AU-ND	14.32 USD	9.47 USD	www.digikey.com
IC2	1	VS1033/VS1053	LQFP48	VS1053B-L / VS1033C-L	16.00 EUR	10.00 EUR	www.vlsi.fi
IC3	1	DS18B20	uSOP-8	DS18B20U+	2.57 USD	2.19 USD	www.maxim-ic.com
IC4	1	MAX1811	SO-8	MAX1811ESA+	2.91 USD	2.06 USD	www.maxim-ic.com
IC5	1	LT1932	SOT23-6	LT1932ES6#TRMPBFCT-ND	4.17 USD	3.35 USD	www.digikey.com
IC6	1	LT1521-3.3	uSOP-8	LT1521CMS8-3.3#PBF-ND	4.63 USD	2.35 USD	www.digikey.com
IC7	1	LM2937-5	SOT223	LM2937IMP-5.0CT-ND	1.83 USD	1.00 USD	www.digikey.com
IC8	1	LT1844-1.8/2.5	SOT-23	LTC1844ES5-*1.8/2.5*#TRMPBFCT-ND	2.88 USD	2.31 USD	www.digikey.com
Òptica							
-	1	LCD Nokia 6610	-	par-lcd6610	19.76 EUR	19.76 EUR	www.jelu.se
LED1	1	LED IR, 5mm	5mm	516-1706-ND	0.33 USD	0.20 USD	www.digikey.com
LED2	1	LED SMT Blau	0603	511-1589-1-ND	0.59 USD	0.23 USD	www.digikey.com
Oscil·ladors							
Q2	1	8Mhz	6x3.6mm	535-9628-1-ND	1.19 USD	0.68 USD	www.digikey.com
Q1	1	12.288Mhz	5x3.2mm	535-9117-1-ND	1.38 USD	0.88 USD	www.digikey.com
Q3	1	32768Khz	3.2x1.5mm	631-1002-1-ND	2.07 USD	1.18 USD	www.digikey.com
Mecànics							
LCD1	1	Connector LCD	HS-DF23	PAR-LCDCON10	0.95 EUR	0.95 EUR	www.jelu.se
AU1	1	Audio Jack	3.5mm	CP-3523SJCT-ND	0.87 USD	0.58 USD	www.digikey.com
DOCK1	1	Dock (Receptor)	ST60-18P	H11363CT-ND	1.98 USD	1.22 USD	www.digikey.com
-	2	Dock (Conn.)	ST40X-18S	H11361-ND	9.10 USD	5.64 USD	www.digikey.com
SD1	1	Socket microSD	-	WM19079CT-ND	3.12 USD	1.92 USD	www.digikey.com
SL1	1	Tecla bloqueig	-	EG1847-ND	1.07 USD	0.77 USD	www.digikey.com
SW6-7	2	Tecles reset	-	CKN9104CT-ND	1.02 USD	0.64 USD	www.digikey.com
SW1-5	5	Tecles control	-	SW423CT-ND	2.25 USD	1.60 USD	www.digikey.com
Semiconductors							
DSK1	1	Díode Schottky, 40V 0.5A	SOD123	MBR0540T1GOSCT-ND	0.26 USD	0.18 USD	www.digikey.com
T1	1	Transistor NPN	SOT-23	MMBT2369ACT-ND	0.14 USD	0.08 USD	www.digikey.com
DZ1	1	Díode Zeners. 24V	SOT-23	BZX84C24-FDICT-ND	0.44 USD	0.20 USD	www.digikey.com
Varis							
JP1	1	Bateria Li-Ion 850mAh	62x35x3.8mm	383562	6.95 USD	5.80 USD	www.all-battery.com
L1	1	Inductor 6.8uH	1008	PCD1238CT-ND	0.27 USD	0.18 USD	www.digikey.com

Resistències							
R1/21	2	100KΩ	0603	P100KGCT-ND	0.16 USD	0.08 USD	www.digikey.com
R2	1	10Ω	0603	P10GCT-ND	0.08 USD	0.04 USD	www.digikey.com
R3/4	2	20Ω	0603	P20GCT-ND	0.16 USD	0.08 USD	www.digikey.com
R5/11/13/15/16/19/20	7	10KΩ	0603	P10KGCT-ND	0.56 USD	0.28 USD	www.digikey.com
R6	1	4K7Ω	0603	P4.7KGCT-ND	0.08 USD	0.04 USD	www.digikey.com
R7	1	1Ω	0603	311-1.00HRCT-ND	0.08 USD	0.04 USD	www.digikey.com
R8	1	220KΩ	0603	P220KGCT-ND	0.08 USD	0.04 USD	www.digikey.com
R9	1	330KΩ	0603	P330KGCT-ND	0.08 USD	0.04 USD	www.digikey.com
R10	1	390Ω	0603	P390GCT-ND	0.08 USD	0.04 USD	www.digikey.com
R12	1	1K5Ω	0603	P1.5KGCT-ND	0.08 USD	0.04 USD	www.digikey.com
R14	1	33KΩ	0603	P33KGCT-ND	0.08 USD	0.04 USD	www.digikey.com
R17/18	2	0Ω	0805	P0.0ACT-ND	0.16 USD	0.08 USD	www.digikey.com
R22	1	1MΩ	0603	P1.0MGCT-ND	0.08 USD	0.04 USD	www.digikey.com
Condensadors							
Ceràmics							
C1/2	2	33pF	0805	PCC330CGCT-ND	0.10 USD	0.06 USD	www.digikey.com
C3/9/14/16/17/19-22/24-26/28-30	15	100nF	0805	PCC1812CT-ND	1.35 USD	0.75 USD	www.digikey.com
C4	1	47nF	0805	PCC1808CT-ND	0.12 USD	0.08 USD	www.digikey.com
C5/6	2	10nF	0805	495-1940-1-ND	0.30 USD	0.22 USD	www.digikey.com
C7/8	2	18pF	0805	PCC180CNCT-ND	0.20 USD	0.14 USD	www.digikey.com
C12/32	2	1uF	0805	PCC1849CT-ND	0.14 USD	0.10 USD	www.digikey.com
Electrolítics							
C10/23/27/31	4	10uF	2012	565-2404-1-ND	0.60 USD	0.56 USD	www.digikey.com
C18/33	2	10uF	3216	511-1473-1-ND	0.56 USD	0.50 USD	www.digikey.com
C11/13	2	4.7uF	3216	511-1442-1-ND	0.56 USD	0.50 USD	www.digikey.com
				SUBTOTAL (USD)	72.03 USD	48.50 USD	
				SUBTOTAL (EUR)	36.71 EUR	30.71 EUR	
				TOTAL (EUR)	**85.71 EUR**	**63.70 EUR**	

[1] Preu en cas de fabricar-ne 10 unitats

[2] Preu en cas de fabricar-ne 100 unitats

1EUR = 1.47USD

** Preus totals per cada unitat del reproductor*

Circuit imprès

Fabricant	Gold Phoenix PCB Co. Ltd.
Pàgina web	www.goldphoenixpcb.biz

Característica	Estàndard (1pan, 10u)	Professional (100u)
- 2 Capes, panell de 650cm^2	99.00 USD	70.00 USD
- Serigrafia 2 cares	20.00 USD	6.00 USD
- Compliment del RoHS	10.00 USD	10.00 USD
TOTAL (USD)	129.00 USD	86.00 USD
TOTAL (EUR)	87.76 EUR	58.50 EUR
PREU UNITAT (EUR)	**8.78 EUR**	**5.85 EUR**

1EUR = 1.47USD

Materials varis

Producte	Preu	Botiga
Materials comuns		
Flux JCB per soldar	4.50 EUR	Zeners Girona
Estany, carret 100gr, 1mm diàm.	5.00 EUR	Zeners Girona
Fixador Metacril·lat Acrifix 192	9.20 EUR	Servei Estació Girona
TOTAL MATERIAL COMÚ	18.70 EUR	
PREU UNITARI (10u)	**1.87 EUR**	
PREU UNITARI (100u)	**0.19 EUR**	
Materials unitaris		
Metacril·lat blanc (5x5cm)	2.00 EUR	Servei Estació Girona
Metacril·lat transparent (15x15cm)	10.00 EUR	Servei Estació Girona
TOTAL (10u i 100u)	**12.00 EUR**	

** Una unitat de material comú és suficient per crear 10 o 100 reproductors*

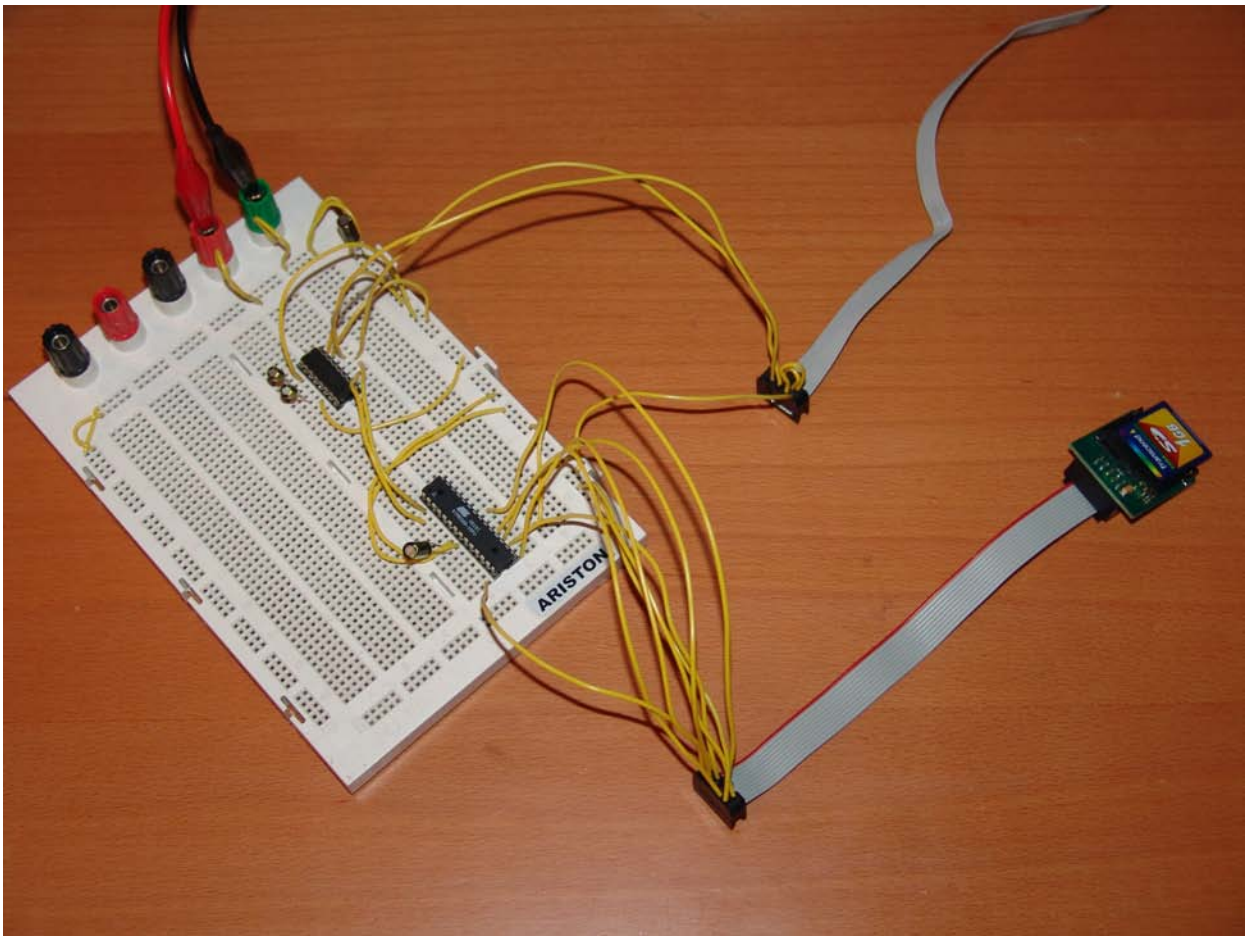
TOTAL

Concepte	Preu unitat (10u.)	Preu unitat (100u.)
Components	85.71 EUR	63.70 EUR
Circuit imprès	8.78 EUR	5.85 EUR
Materials Varis	13.87 EUR	12.19 EUR
Mà d'obra	∞	
TOTAL	**108.36 EUR**	**81.74 EUR**

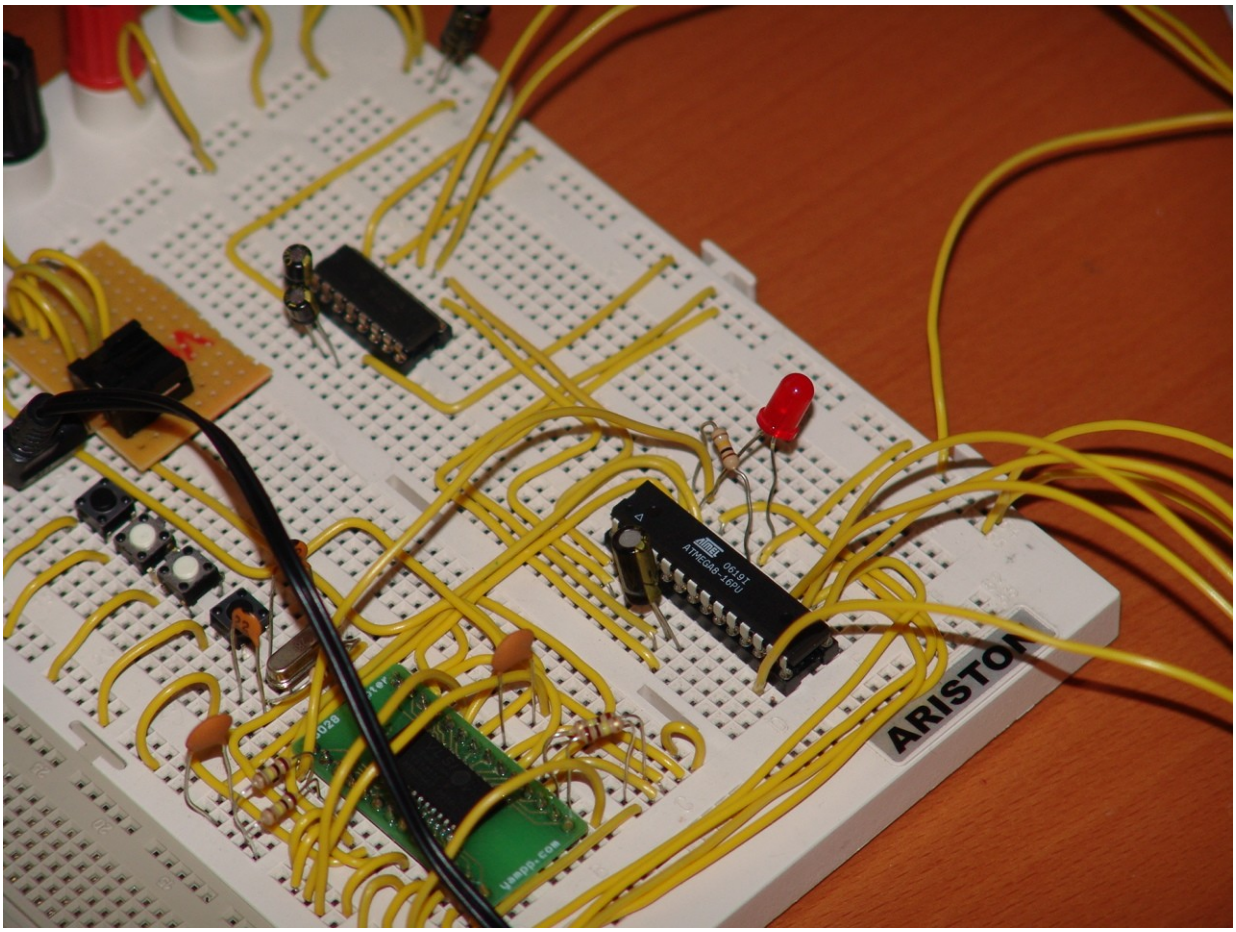
Els totals calculats no inclouen les despeses d'enviament del material

Annex 4

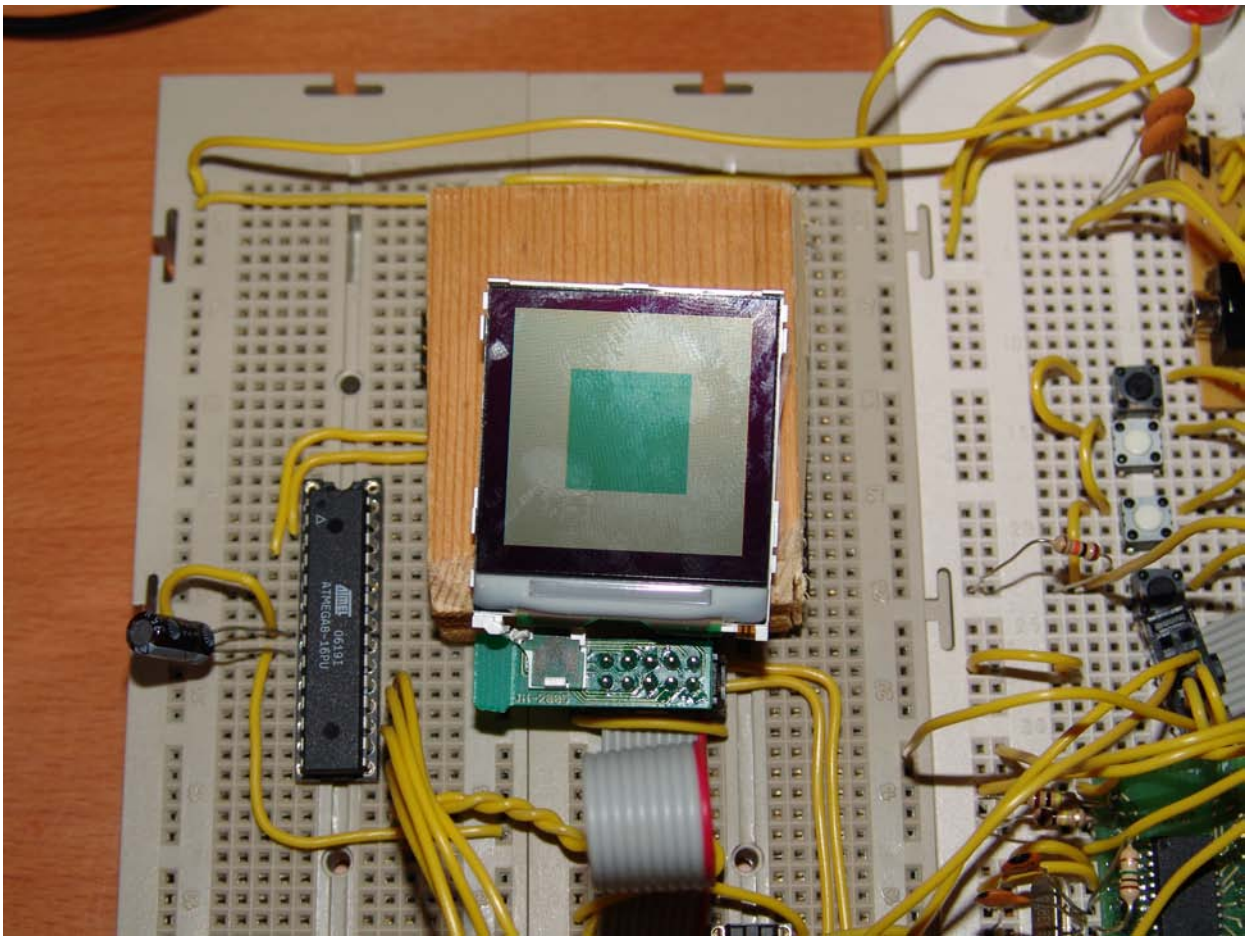
Fotografies



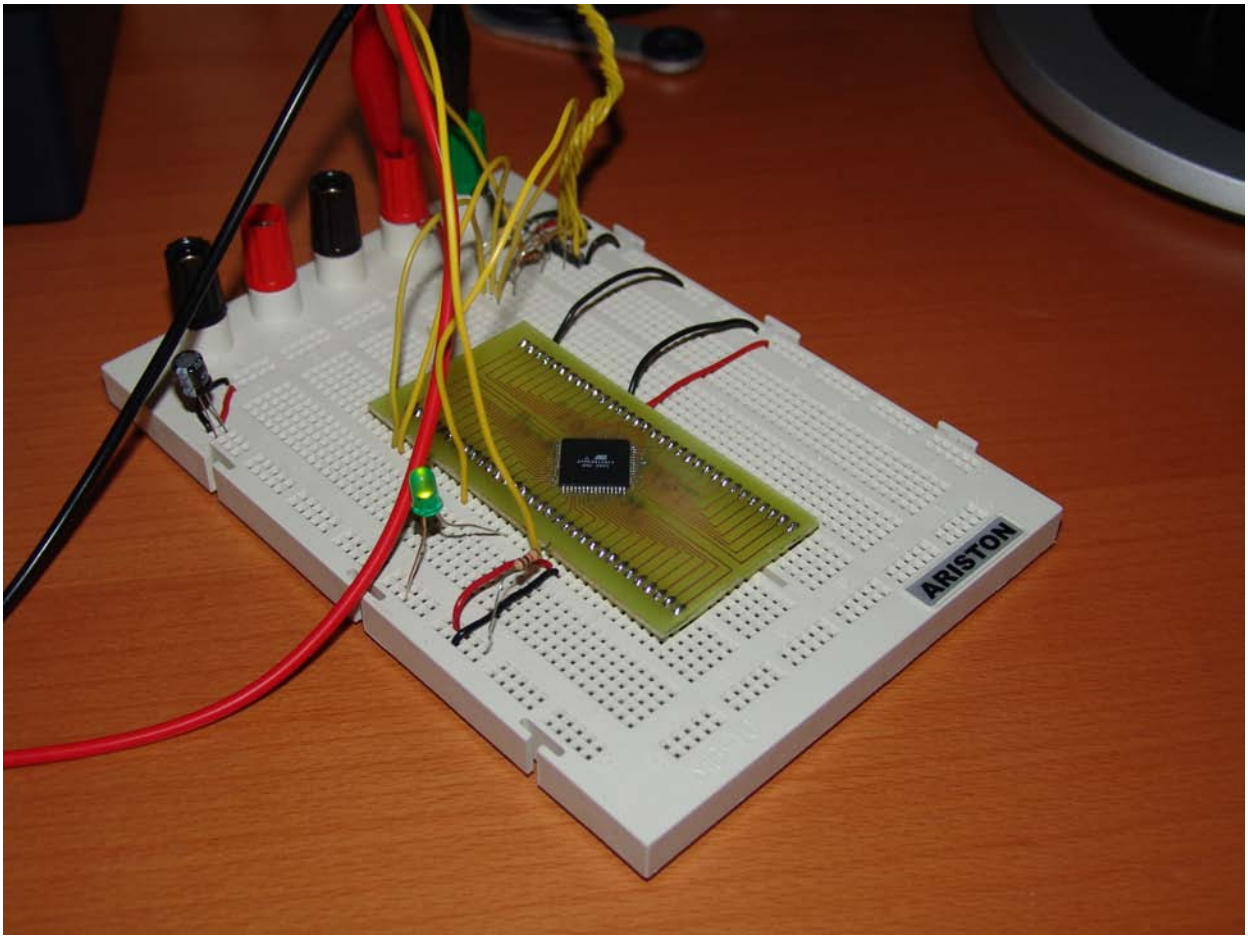
Març 2007: Es llegeixen dades de la targeta SD correctament



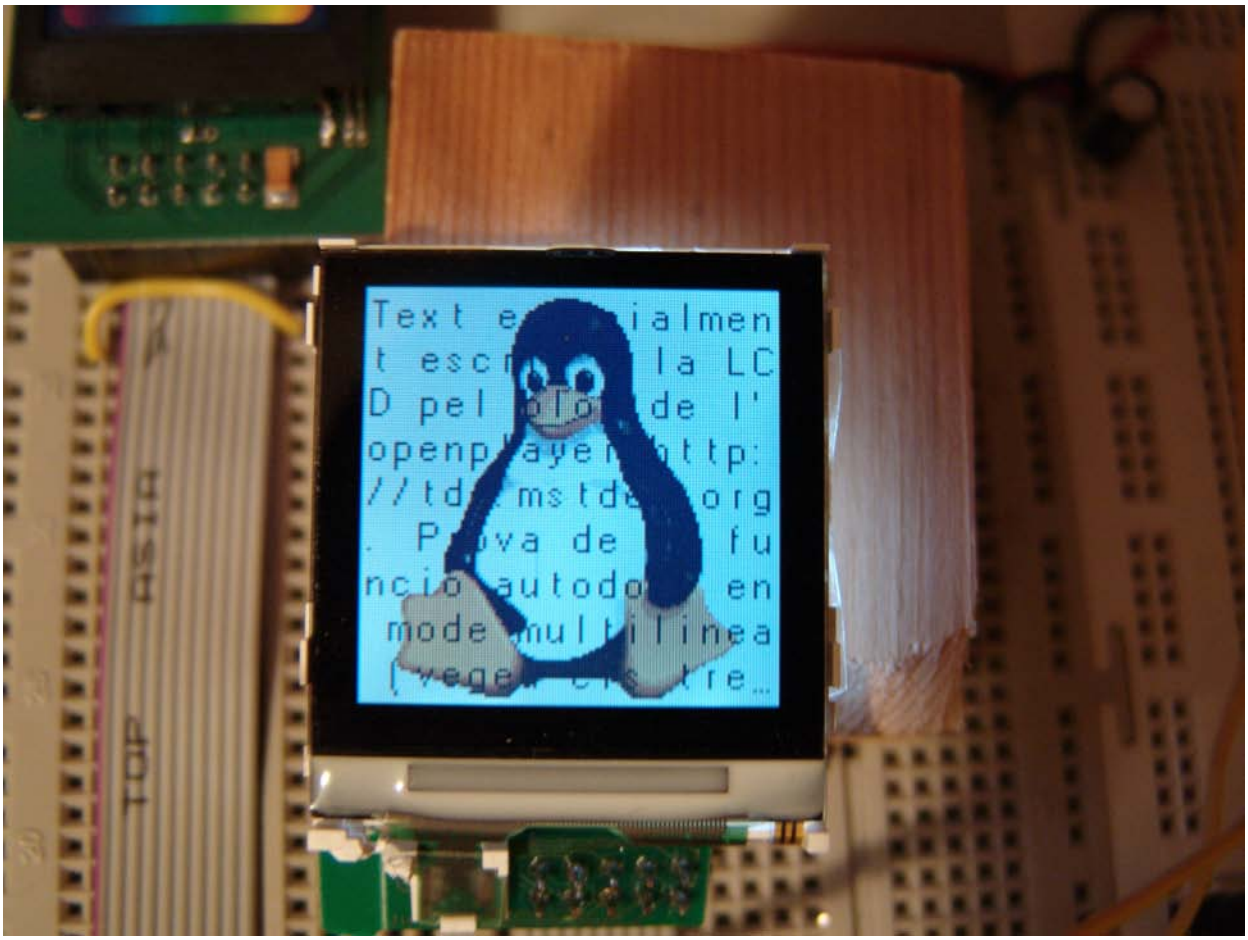
Abril-Maig 2007: Es crea l'intèrpret dels sistemes de fitxers FAT16/32 i el controlador del descodificador, cosa que possibilita que comencin a sonar les primeres cançons



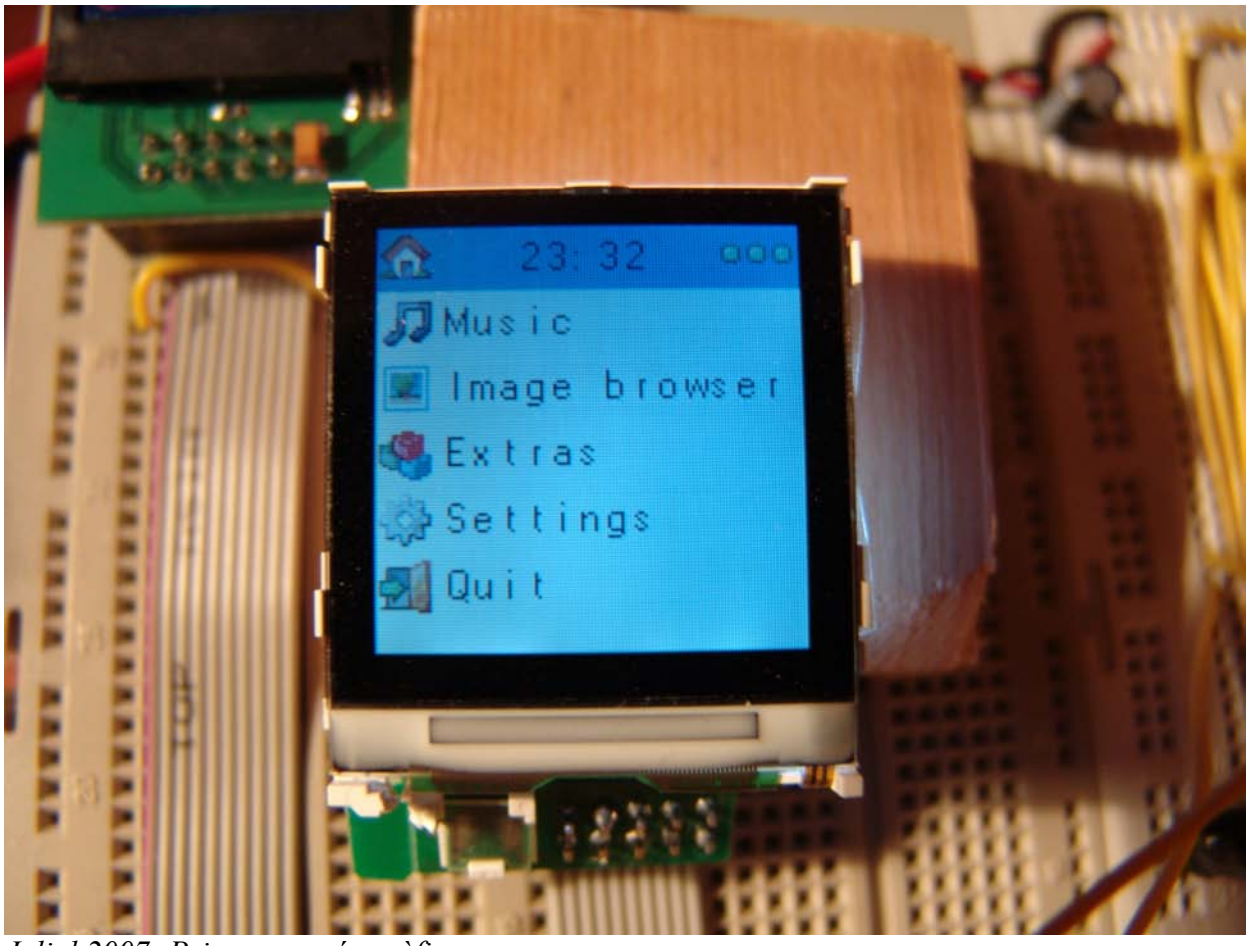
Juny 2007: El controlador bàsic de la LCD comença a funcionar



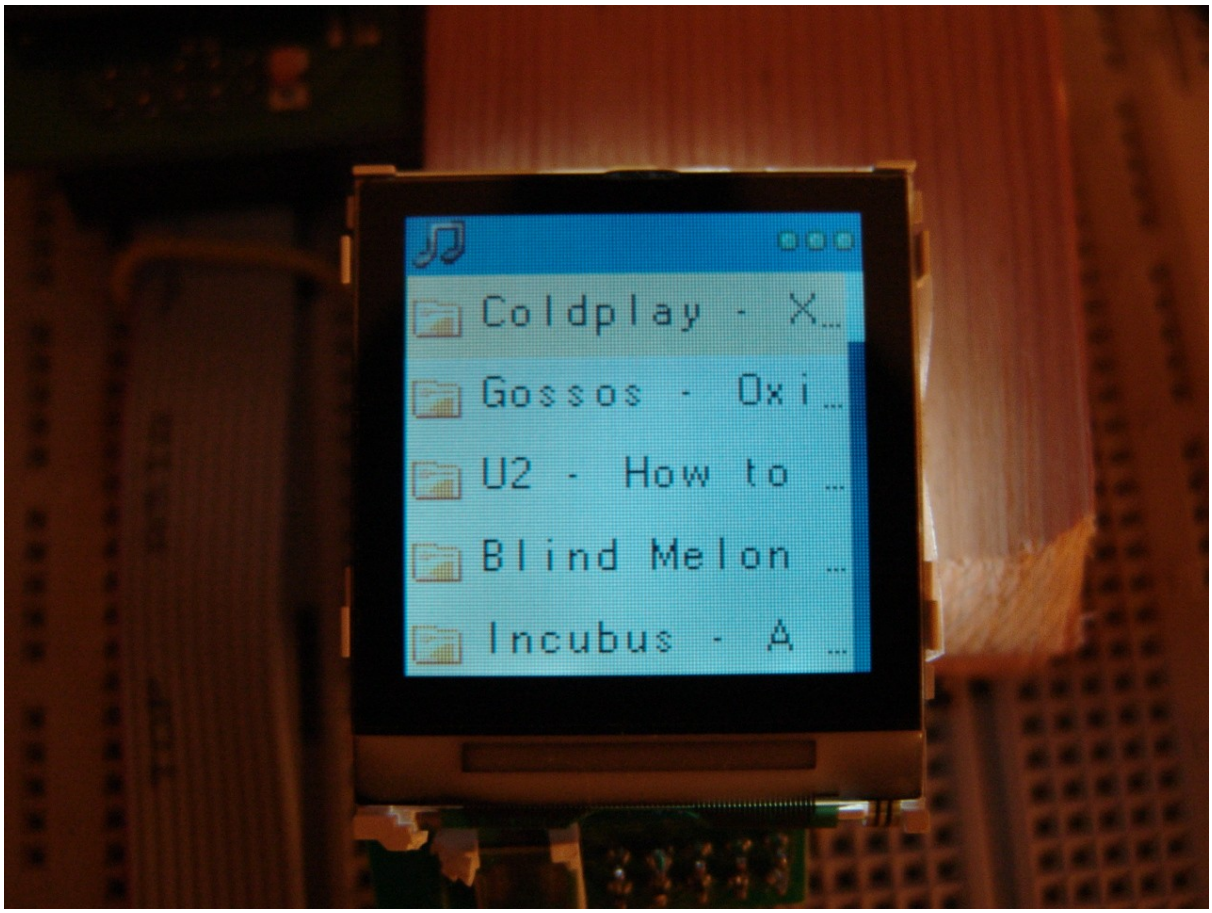
Juny 2007: Es fa necessari el canvi al nou microcontrolador (Atmega1281). A la foto, el primer firmware executat satisfactòriament sota el nou microcontrolador (LED intermitent)



Juny-Juliol 2007: Ja amb el nou microcontrolador, creació de les funcions gràfiques bàsiques per la pantalla (text, càrrega d'imatges des de la targeta SD, etc...)

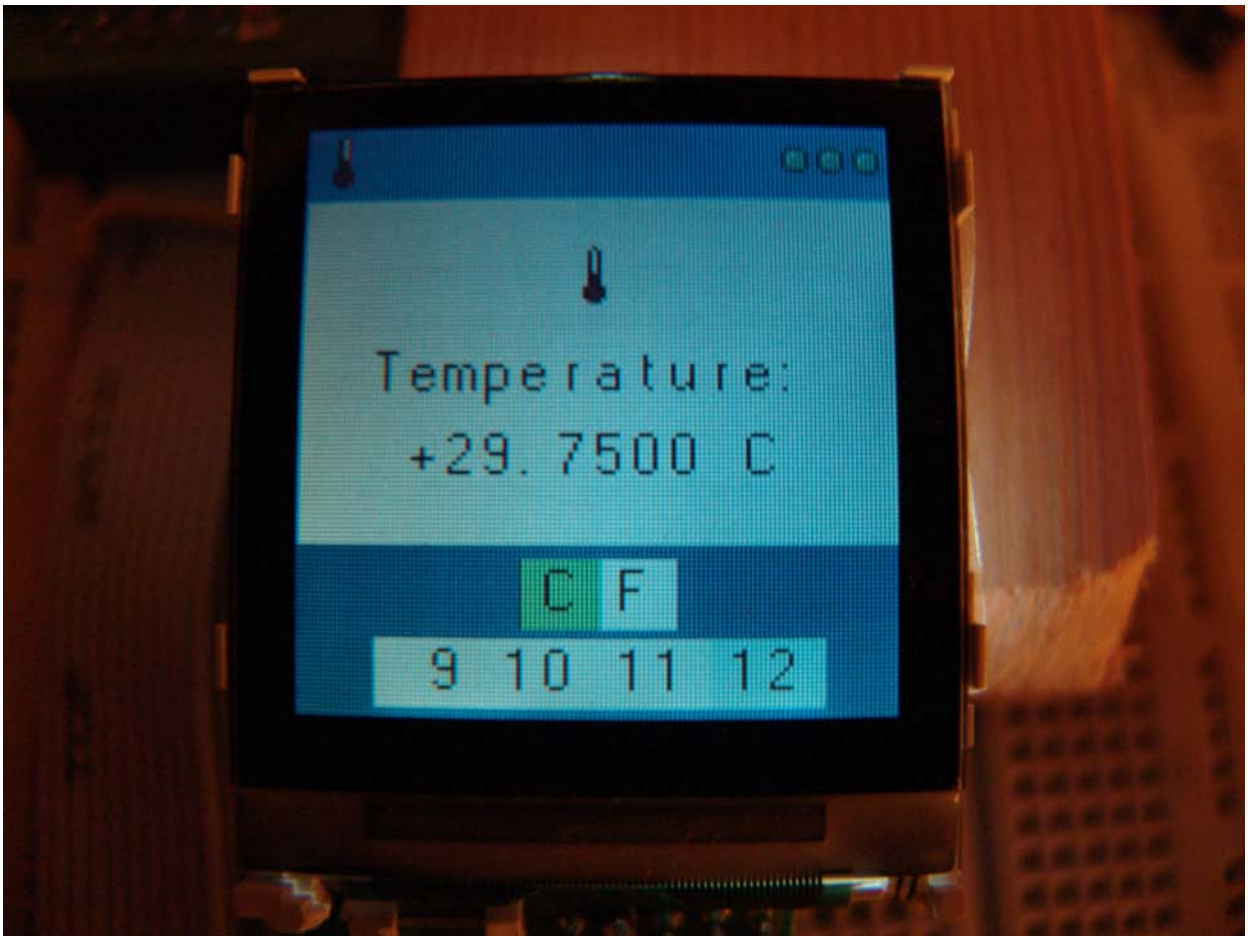


Juliol 2007: Primers menús gràfics

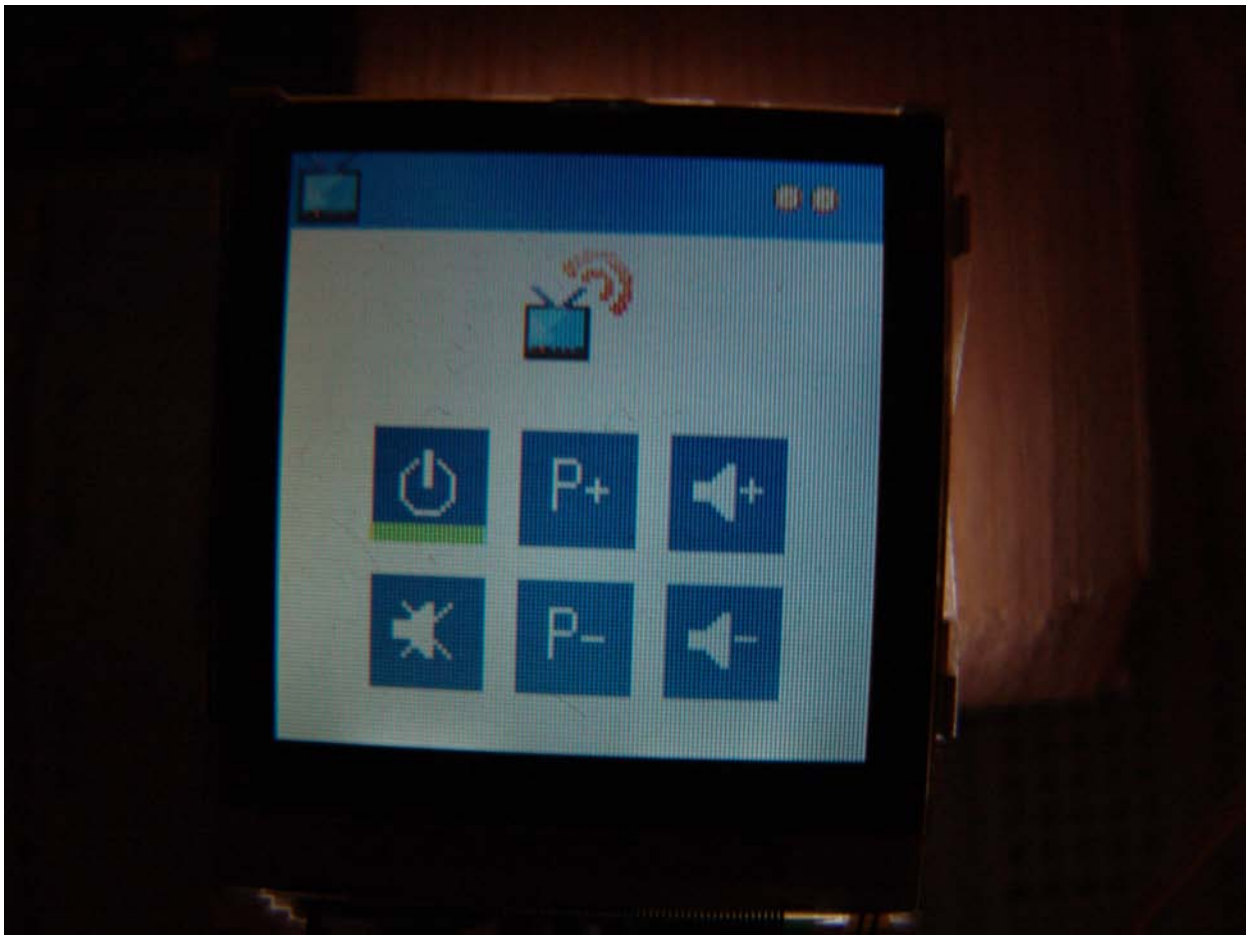


Agost 2007: Creació del navegador de fitxers gràfic, que servirà per poder navegar entre la música desada a la targeta SD

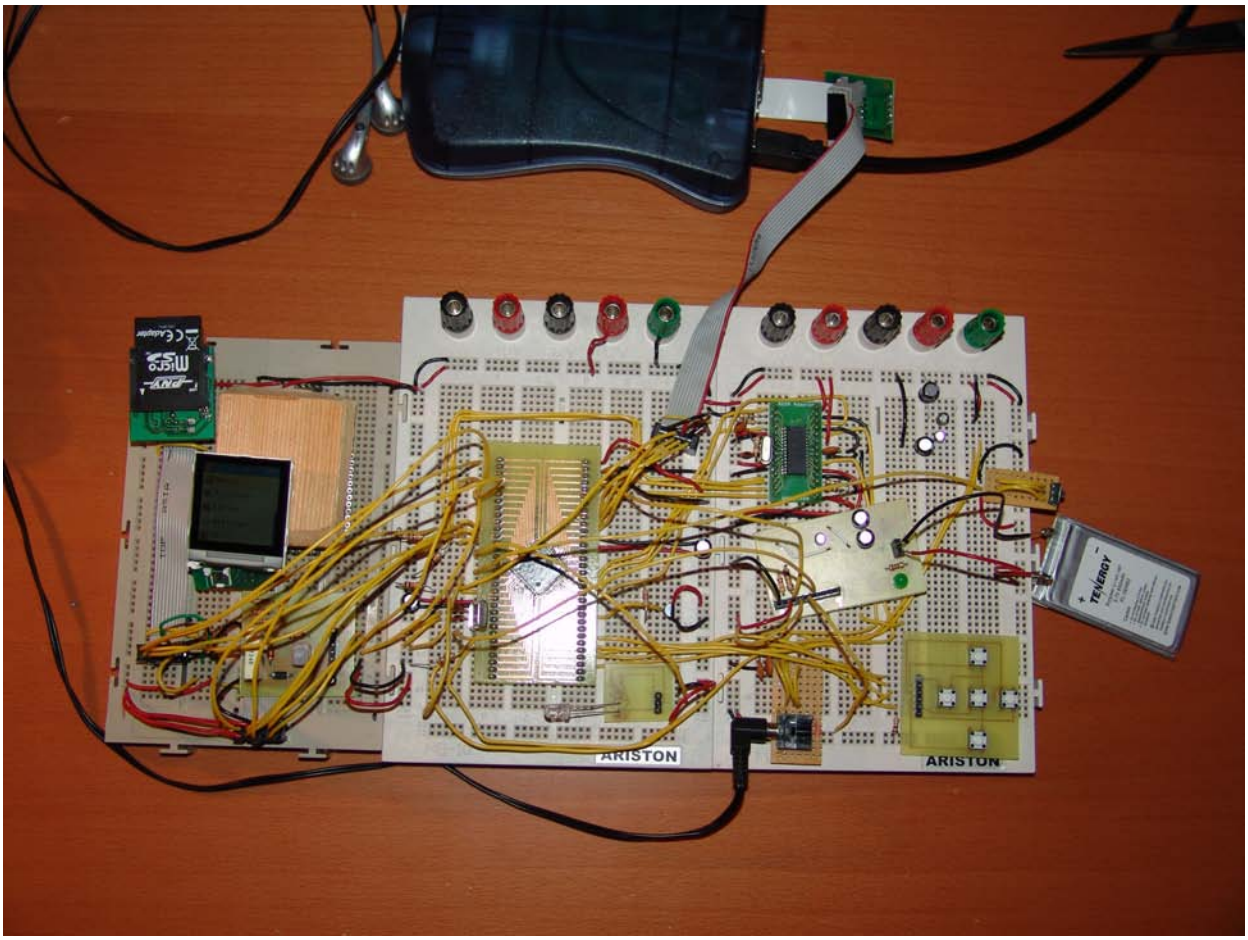
Agost 2007: Creació del reproductor de música gràfic, incloent suport per caràtules i ID3 tags



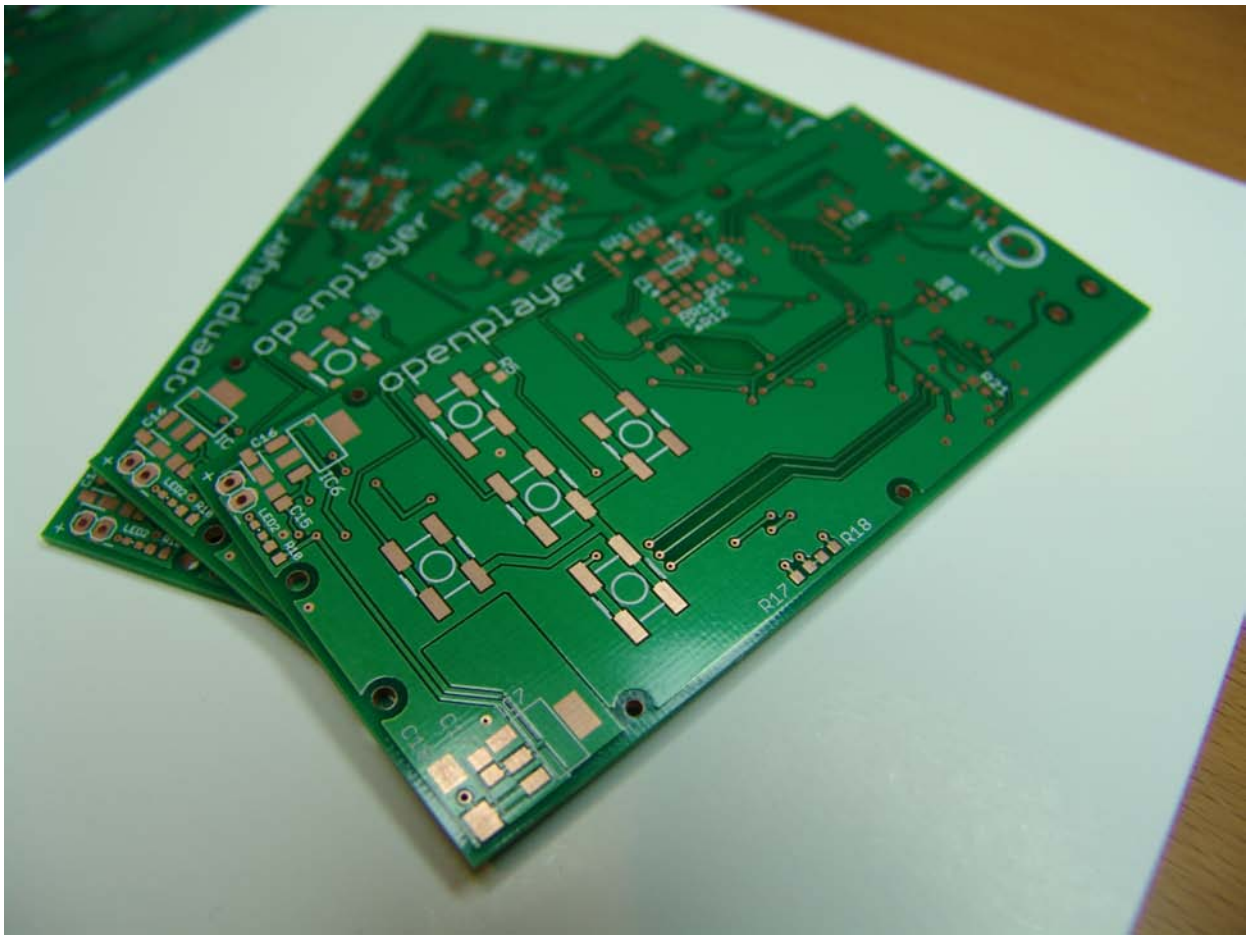
Septembre 2007: Implementació del termòmetre digital incloent una aplicació gràfica per visualitzar la temperatura



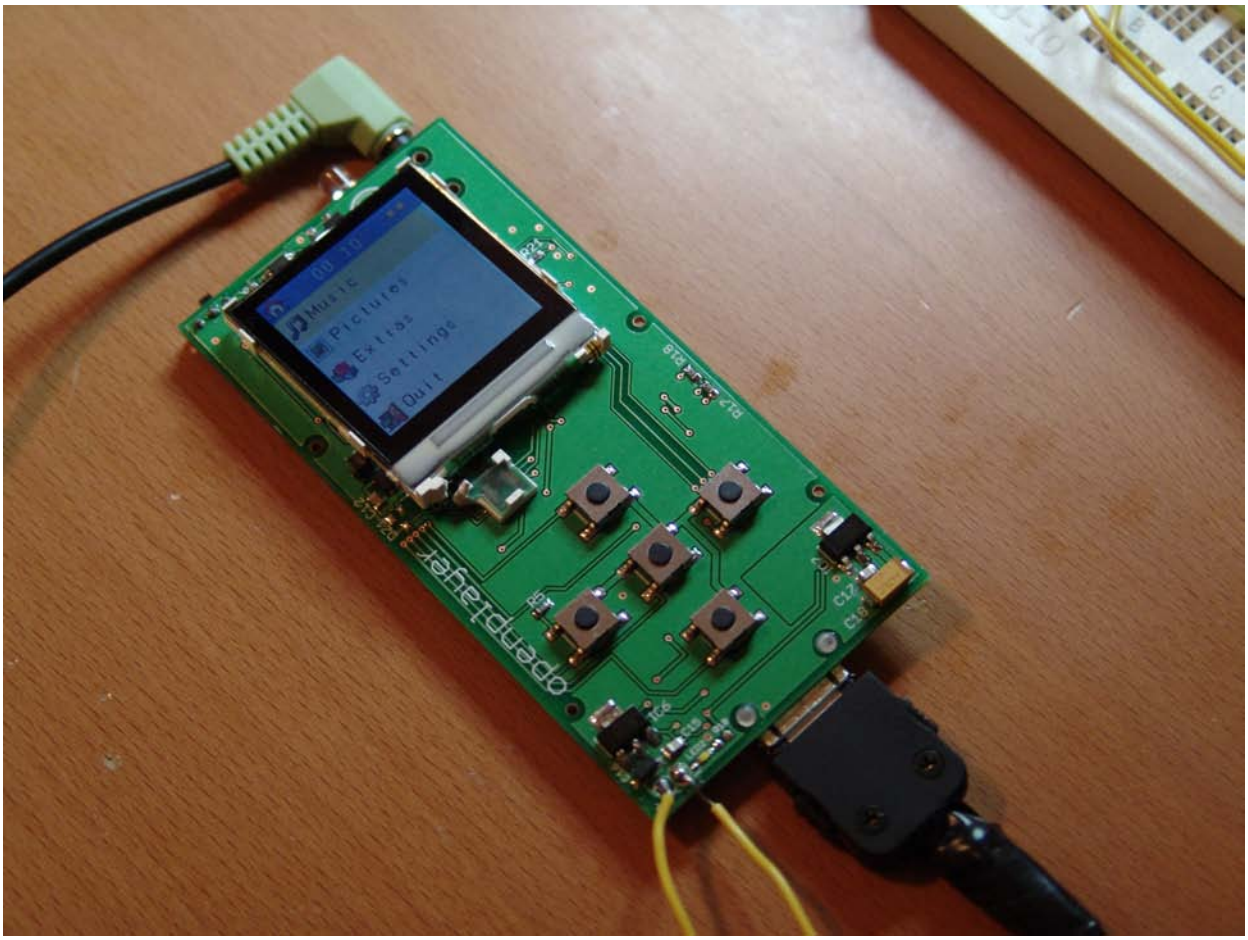
Octubre 2007: Creació del comandament de televisió



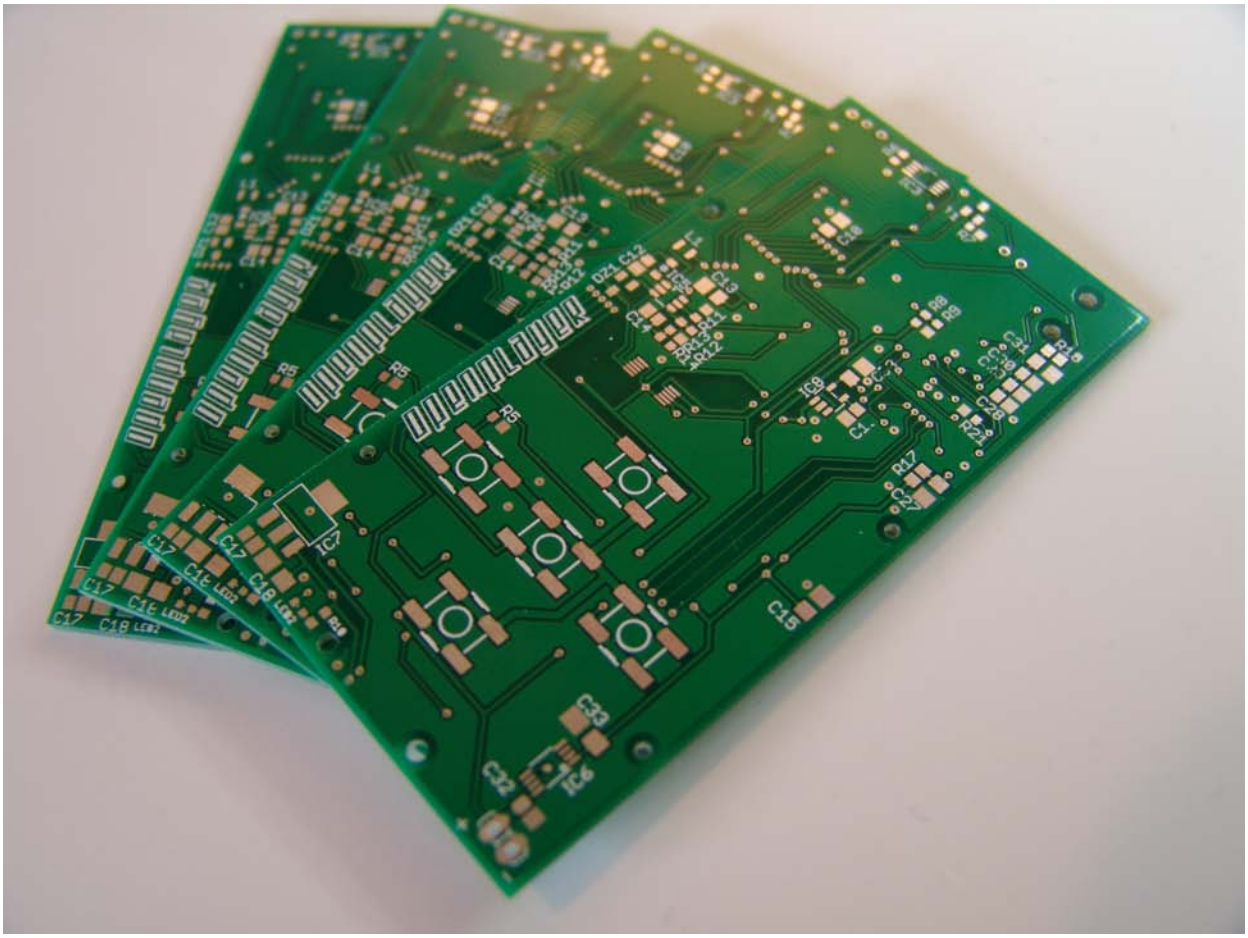
Novembre 2007: Circuit final abans de crear la placa de circuit imprès, inclòs el carregador de bateria



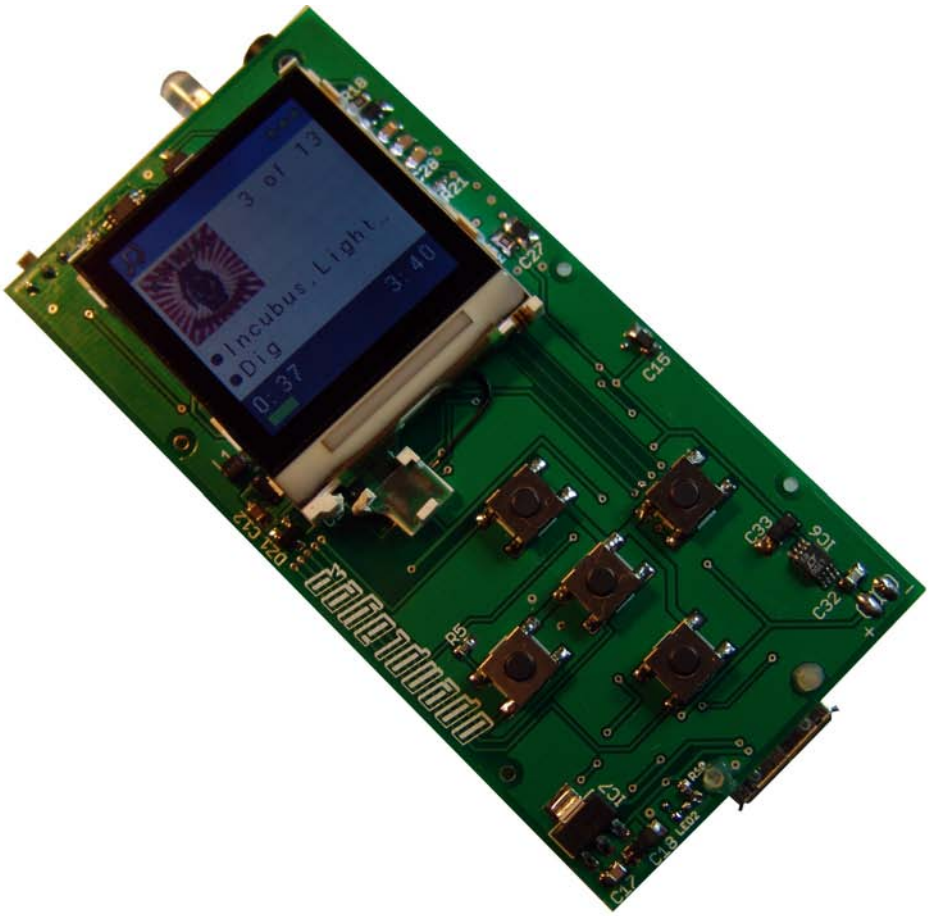
Desembre 2007: Les primeres plaques (revisió 1.0), una realitat després de moltes hores en el seu disseny



Desembre 2007: La primera versió de l'openplayer portàtil ensamblada, la qual conté alguns errors de disseny que provoquen problemes d'estabilitat en reproduir música



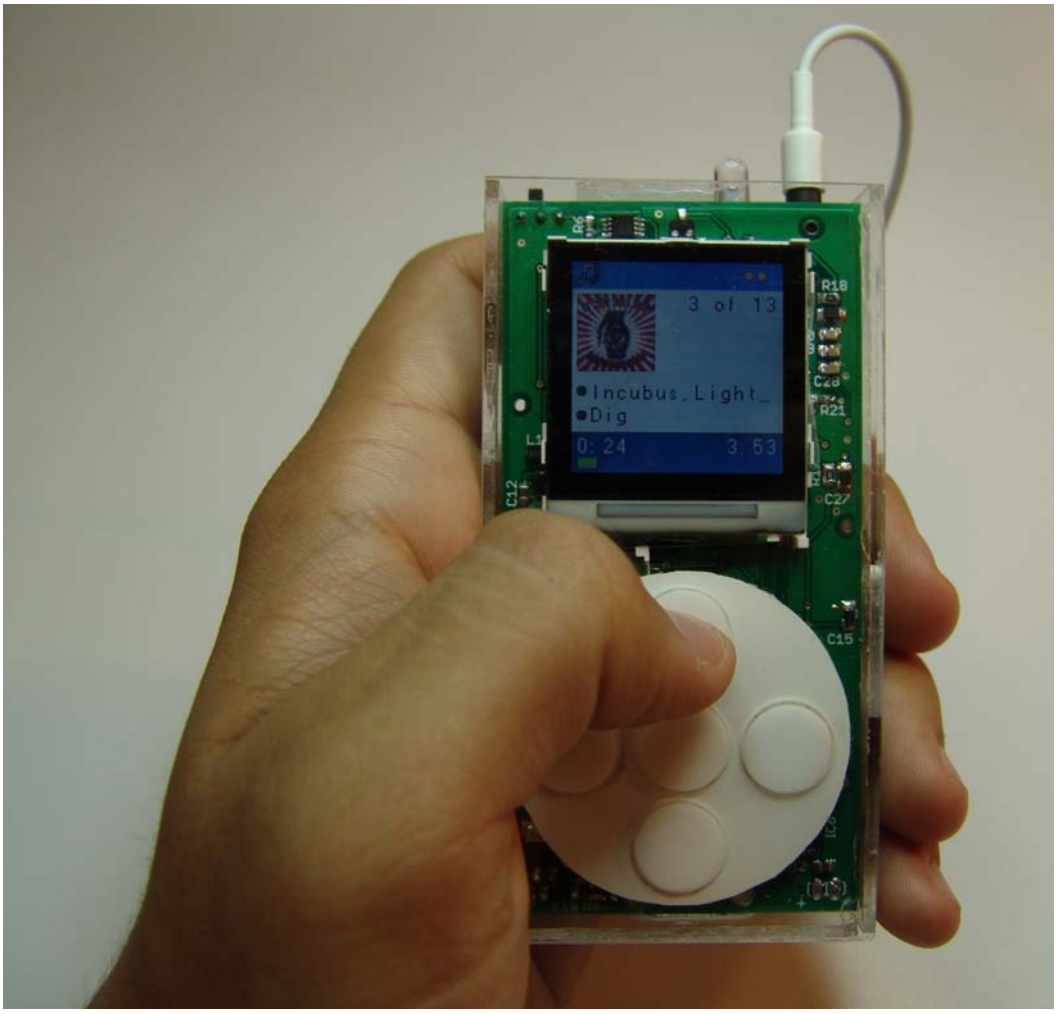
Gener 2008: Arriben les plaques noves (revisió 1.1) amb les correccions fetes



Gener 2008: La nova placa ensamblada, sense errors i amb un funcionament totalment estable

Gener 2008: La carcassa de metacril·lat junt amb el reproductor final

Gener 2008: La carcassa de metacril·lat amb el reproductor final (part de darrera)



Gener 2008: Edició totalment acabada reproduint música

Annex 5

Contingut digital

1. Introducció

Aquest annex està format bàsicament per documents o fitxers que no es poden imprimir en paper ja que el seu tamany és molt gran o també perquè és més pràctic visualitzar-los mitjançant un ordinador. Els documents es troben inclosos dins un DVD que només caldrà introduir en un ordinador i obrir el fitxer "index.html", una petita pàgina web on es podrà accedir al contingut del DVD de manera ordenada com també obtenir-ne informació.

2. Resum del contingut

2.1. Codi

La part més important de l'openplayer és el codi i tot i que en aquest treball no s'hagi aprofundit en ell. Es podrà accedir i utilitzar el codi segons la llicència GPLv3, inclosa dins el mateix DVD.

2.2. Esquemes i circuits

A l'Annex 1 s'inclouen imprimits els esquemes i circuits de l'openplayer. Tot i això es recomana visualitzar-ne les versions digitals per ontenir-ne major detall. En el DVD s'inclouen els originals (editables), els fitxers Gerber/Excellon, els fitxers panelitzats, etc. També s'inclou la llibreria de components de l'openplayer que vaig crear pel programa Eagle i els scripts per generar els fitxers Gerber, Excellon i de panelització. Alguns d'aquests fitxers requereixen utilitzar un programa per visualitzar-los, inclosos també en el DVD (apartat 2.5)

2.3. Fulls d'especificacions i documents tècnics

Gairebé tots els components que formen l'openplayer tenen un full d'especificacions (coneguts com a *datasheets*). En el DVD s'inclouen tots els dels components inclosos a l'openplayer (microcontrolador, pantalla, reguladors de voltatge...).

Durant el procés de programació també vaig utilitzar diversos documents on s'expliquen detalls tècnics d'algun tema (ex. l'estructura dels sistemes de fitxers FAT16/32). Tot i ser únicament d'ús tècnic els he inclòs per aquelles persones interessades en consultar més detalls d'un tema concret.

2.4. Fotografies i vídeos de l'evolució de l'openplayer

Tot i que a l'annex anterior es mostren algunes fotografies del procés de construcció, en el DVD s'ha inclòs la col·lecció completa. A la pàgina web del DVD s'inclou una petita galeria on visualitzar totes les fotografies. També s'inclouen alguns vídeos on es poden observar funcionalitats de l'openplayer en funcionament fetes en el moment de la seva creació.

2.5. Programari

Alguns documents com els esquemes originals o els fitxers Gerber requereixen utilitzar uns programes determinats per obrir-los, i per tant he decidit incloure'ls al DVD. També s'inclouen els programes descrits en el Capítol 2 que corresponen als programes utilitzats en el procés de programació (gcc, avrdude, eclipse...). Alguns programes s'inclouen per diverses plataformes (Linux, MacOS X i Windows) i en altres s'inclou el codi per poder-los compilar a la plataforma que desitgi l'usuari. La llista completa de programes inclosos en el DVD és la següent:

- Eagle 4.16r2
- Gerbv 1.0.3

- gerbmerge 1.1
- GCC 4.2.1 (inclosos *patchs* específics per la plataforma AVR)
- GNU Binutils 2.17 (inclosos *patchs* específics per la plataforma AVR)
- avr-libc 1.46
- GDB 6.6 (inclosos *patchs* específics per la plataforma AVR)
- avrdude 5.4
- Avarice 2.6
- Eclipse 3.3 (+ CDT 4.0 i Zylin)
- GCLCD 3.90

2.6. Llicències

La llicència d'aquest document correspon a la GFDL 1.2 i la del codi la GPLv3. Totes dues llicències es troben dins el DVD.

2.7. Altres documents

Durant el procés de construcció de l'openplayer vaig anar redactant alguns documents tècnics sobre temes que involucraven a l'openplayer. Alguns documents no estan totalment acabats i poden contenir errors, ja que no han estat revisats exhaustivament. Tot aquest conjunt serà polit i unit en els pròxims mesos (potser durant l'estiu) i es podrà descarregar des de la pàgina web http://www.teslabs.com/openplayer. Els títols publicats i inclosos en el DVD són els següents:

- Setting up AVR development environment (for Linux, MacOS X and Windows)
- Using DS18B20 digital temperature sensor on AVR microcontrollers
- EFS: EEPROM File System
- Sistemes de fitxers FAT16/32
- Descodificador MP3 VS1011e
- Operacions amb targetes SecureDigital

També cal dir que s'inclou una còpia del treball en PDF dins el DVD.

3. DVD