What I Did On My Summer Vacation

Or… How to build your own MP3 player

Frank Van Hooft
5 Oct 2003

Agenda

- MP3 Player Requirements
- Hardware Architecture
- Software Architecture
- Next Steps

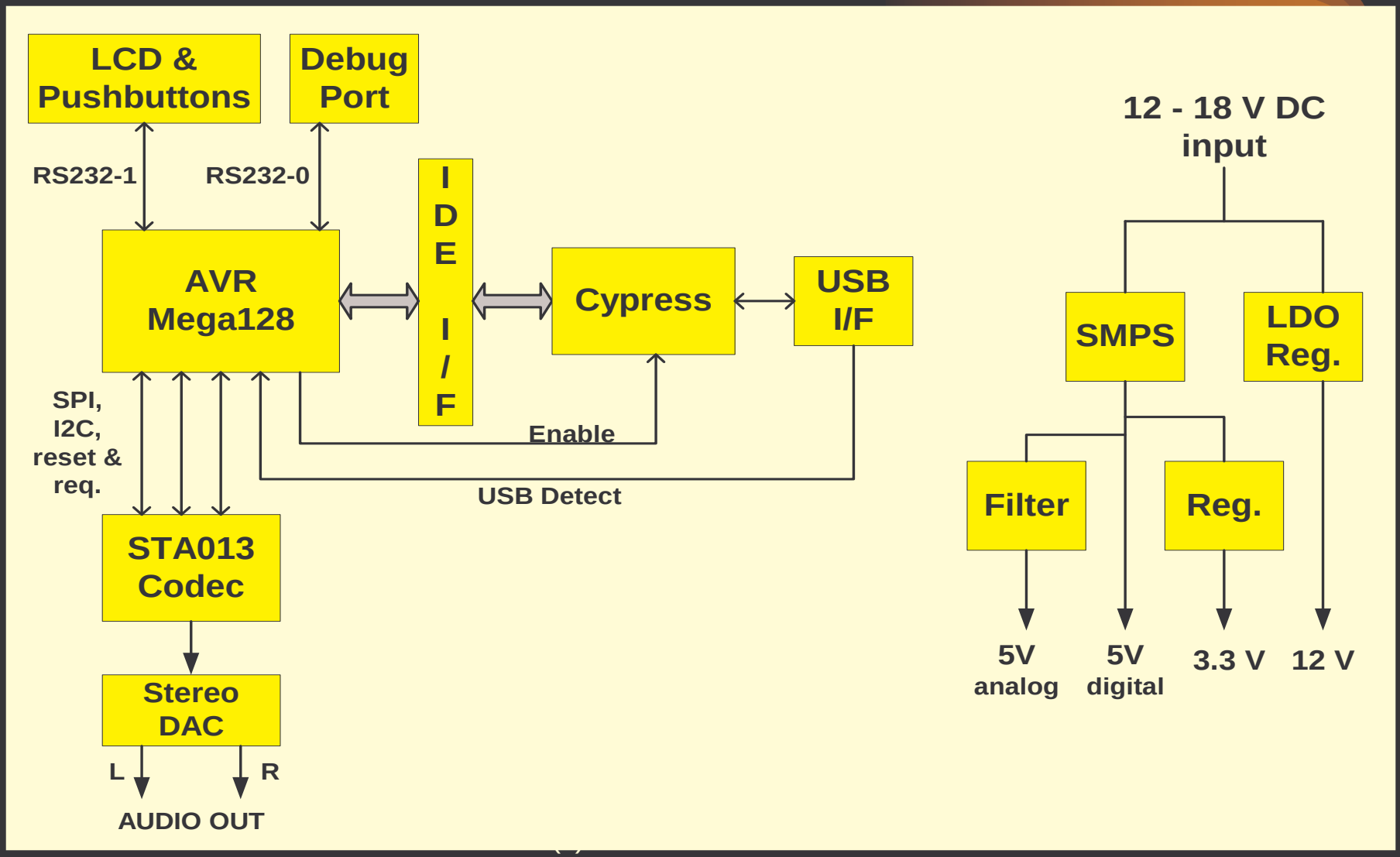
MP3 Player Requirements

- Capable of handling > 1000 songs
- Standalone operation – no PC required
- Simple & fast mechanism for song upload
- Capable of acting as a home stereo component
- Capable of functioning in a car
- Relatively easy to develop

These requirements resulted in…

- Hard-disk based. FAT-32 file system.
- 3.5” disks preferred. Max 137 GB.
- Must handle fragmented files & directories.
- 8 x 24 LCD display and push-buttons.
- USB 2.0 interface.
- Line-level audio outputs.
- 12V DC input with surge suppressors (for automotive environment).

Hardware Block Diagram



Power Supply

- +12 to +18 V DC input
- On-board Transzorb surge suppressor & fuse
- +12V for disk drive provided by LDO regulator
- +5V digital for disk drive & AVR via SMPS
- +5V analog for stereo DAC & filters
- +3.3V for Cypress & STA013 from linear regulator

Atmel AVR Mega128

- The brains of the player.
- IDE interface implemented by GPIO pins:
  - bit-bashing!
  - uses PIO Mode 0/1/2 for simple accesses
    (DMA transfers not supported, nor required)
- ST codec interfaced via two “serial” ports:
  SPI and I^2C. Also controls codec reset.
- RS232 port 0 for debug
- RS232 port 1 for LCD display & pushbuttons
- Monitors USB status & controls USB enable

STA013 MP3 Codec Chip

- Accepts an MP3 serial data stream over its SPI (serial peripheral interface) port.
- Performs all MP3 decoding operations.
- Generates clocks & digital bitstreams for the Crystal stereo audio DAC (glueless interface).
- Has an I^2C port for initialization, control & status.
- Has a “data request” pin for signalling when it wants more MP3 data.

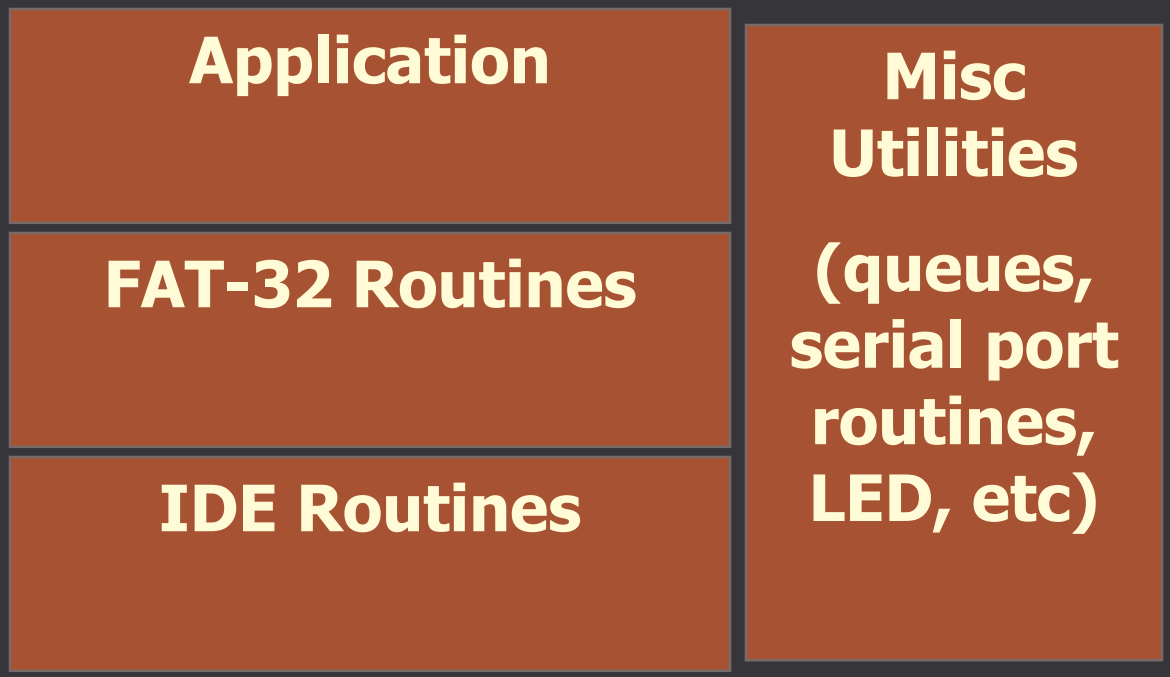
Cypress USB 2.0 Interface

- An intelligent core with embedded USB interface logic. A glueless “bolt-on”.
- Sits on the IDE bus in parallel with the AVR.
- AVR keeps USB disabled unless USB cable plugged in.
- Cypress provides a “mass storage class” reference design, including a Win98 driver.
- Appears as a harddrive in Windows Explorer.
- Fast! Suitable for very large file transfers.

Software Considerations

- The Mega128 has 4 kB of RAM.
- 3.5 kB is used for the MP3 data queue.
- Leaving only 512 bytes for the stack, global variables, other queues & buffers, etc!
- The Mega128 has 128 kB of Flash program memory. So *always* use extra program space if it'll reduce data space usage.
- Minimal interrupts.
- Uses a simple "main loop" structure.

Software Stack

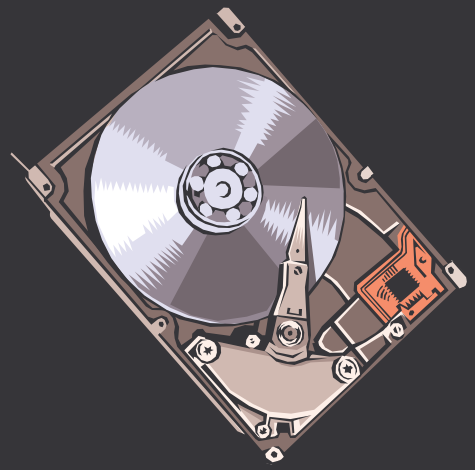


Misc Routines

- Queues, AKA FIFOs, ring-buffers, etc. May be any length; not restricted to powers-of-2.
- Serial port routines allow writing chars, strings, hex numbers, ASCII numbers to the debug & LCD screens. Also handle keypresses.
- Serial port routines control the SPI & I^2C ports for the codec chip interface (incl. init). And monitor & control the USB interface.
- EEPROM read & write routines
- Where would we be without an LED?

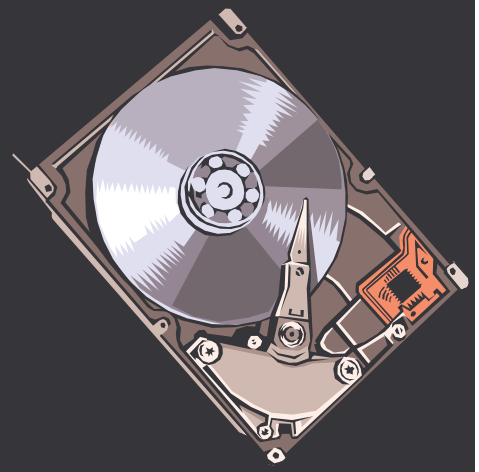
IDE Routines

- Some basic routines:
  - **Initialization**
  - **Reset drive**
  - **8-bit read**
  - **8-bit write**
  - **16-bit read**
- And also a more complex state machine:
  - **SeekSector**



The 137 GB "limit"

- All modern IDE drives support LBA: Logical Block Addressing
- A block is a sector: 512 bytes
- The drive has a 28-bit field for selecting the sector number.
- 2^28 x 512 = 137 GB



FAT32 Disk Structure

FAT32 Routines

- Initialization
- Reading of the drive parameters (cluster size, FAT size, etc.)
- Cluster to sector conversion
- Determine next cluster (ie FAT lookup. Including lookahead function.)
- StreamFile
- FindFile
- GetLongFileName

Application Routines

- PlayControl !
  This state machine…
  - **responds to keypresses,**
  - **controls the finding and then streaming of the MP3 files,**
  - **deals with random & repeat modes,**
  - **displays appropriate things on the LCD.**

- Count MP3 files.
- Format file name string

S/W Development Environment

- Mix of AVR assembly code and C code.
- Uses the GNU C compiler (& linker, assembler, make utility) under DOS.
- PonyProg (shareware) for programming the AVR.
- AVR-Studio (free download) as code editor & ASM code simulator.

- Total of ~6000 lines of code.
  (plus a 2000 line STA013 initialization file.)

Current Status

- Today the MP3 player can:
  - Handle FAT32 disks of varying sizes;
  - Deal with fragmented files & directories;
  - Count the MP3 files in the root directory;
  - Play them in the disk order or randomly;
  - Handle high-bitrate MP3 files;
  - Display the long filename properly formatted;
  - Has basic on/off, start/stop/next functionality.

Next Steps

- Improve the user interface:
  - Add menus
- Add PlayList support
- Add control for STA013 output level
- Add subdirectory support
- Add IR remote control support (hardware already present)

The End…

Thank You!