

ДЕЛАЕМ СЕНСОРНЫЙ ПЛЕЕР НАПОДОБИЕ IPOD TOUCH

STM32

для ЗВУЧНОГО гаджета



Если хочешь не отставать от современных микроконтроллерных тенденций и иметь возможность сделать свой планшет, спутниковую автомобильную сигнализацию или умный дом — пора разобраться с 32-разрядными микроконтроллерами. Сегодня мы посмотрим, как работать с основанным на Cortex M3 контроллером STM32 и сделаем собственный iPodоподобный плеер.

ЧТО НАМ НУЖНО?

Сейчас одни из самых популярных микроконтроллеров на ядре Cortex (на ядре семейства Cortex, кстати, делаются и гаджеты Apple) — микроконтроллеры семейства STM32. Именно с ним мы и будем работать на примере STM32F103.

Но вначале разберемся со средствами разработки. Нам нужна плата с уже впаянным микроконтроллером и максимумом периферии, чтобы в процессе разработки обойтись без паяльника. В качестве отладочной платы я взял на eBay один из клонов платы Nu Fireball (за 76 долларов продается на bit.ly/Q0o4zf). Микроконтроллер на ней стоит STM32F103VCT6 (72 МГц, 256 Кб FLASH, 48 Кб SRAM, 3 АЦП, 2 ЦАП, DMA, FSMC, 3 x USART, 2 x UART, 2 x I2C, 3 x SPI, CAN, USB 2.0 FS и другой фарш). Из периферии на плате расположено следующее богатство:

- MP3/WMA/MIDI-декодер и ADPCM-кодер VS1003 — немаловажная часть нашего проекта, принимает от микроконтроллера данные в цифровом виде и воспроизводит их. Также позволяет реализовать диктофон, оцифровывая данные с микрофона;
- ENC28J60 — Ethernet-контроллер, позволяющий подключить к интернету наш STM32;
- CH376 — чип, реализующий USB-режимы Device/Host и позволяющий подключать SD-карты. Реально полезен из-за поддержки

USB Host режима, так как дает возможность подключать USB-флешки и прочие USB-накопители к микроконтроллеру через SPI либо последовательный интерфейс;

- порт CAN — позволяет подключить нашу плату к шине автомобиля;
- сенсорный резистивный экран 3,2", последовательные порты, 128 Мб NAND Flash, порт RS-485 и куча иных вкусностей.

Отладочную плату мы выбрали, теперь встает вопрос, как защищать микроконтроллер на ней. Можно обойтись ресурсами самой платы, заливая прошивку через последовательный порт (для этого служат джамперы Boot0, Boot1 на плате), но гораздо удобнее пользоваться внутрисхемным отладчиком, который позволит в любой момент видеть все, что творится внутри микроконтроллера. Можно взять JTAG-дебаггер, но гораздо дешевле и целесообразнее будет взять любую из плат семейства STM32 Discovery. Я пользуюсь STM32LDISCOVERY со встроенным ЖК-экраном, предназначенной для разработки портативных устройств с малым энергопотреблением (стоит 22 доллара), но можно взять плату подороже и с более мощным МК — STM32F4DISCOVERY (за 30–35 баксов, например, на все том же eBay: bit.ly/LhOWV1) — на борту микроконтроллер STM32F407 (168 МГц, 1 Мб Flash, 192 Кб RAM, Ethernet, интерфейс камеры, DSP и другое), акселерометр, MEMS-микрофон, ЦАП, USB OTG разъем + примеры с исходными кодами по работе с акселерометром, диктофон/плеер с использованием внешней USB-флешки и так далее. Чтобы соединить отладчик с отладочной платой, удобно использовать джамперный кабель (2 доллара на bit.ly/N95Vip).

Что касается программного обеспечения — тут нет заведомо наилучшего решения, на вкус и цвет каждому свое. Есть бесплатная среда Eclipse/GCC, есть платные IAR и Keil (www.keil.com/demo/eval/arm.htm). Последняя наиболее распространена и, на мой взгляд, куда удобнее остальных. Бесплатная версия позволяет собирать до 32 Кб кода, но с волшебными лекарствами, исцеляющими от этого ограничения, проблем на просторах инета нет.

Залить прошивку в микроконтроллер можно через среду разработки либо отдельной утилитой STM32 ST-LINK Utility.

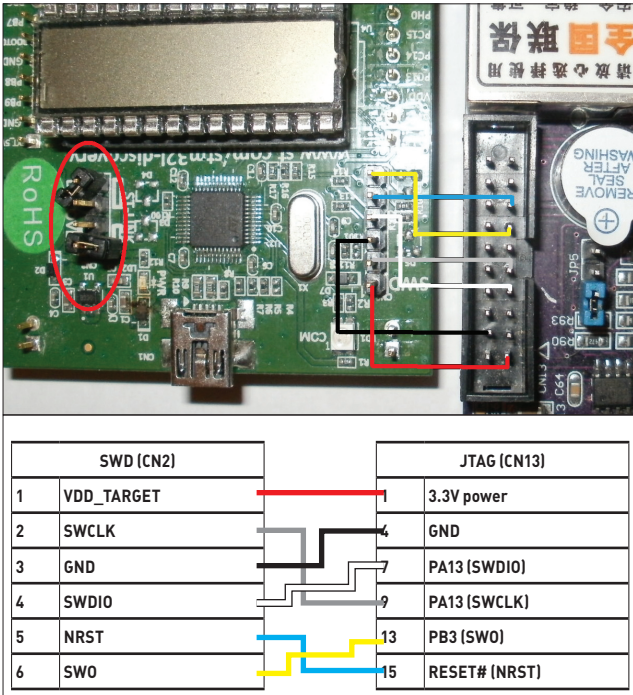


Рис. 1. Схема подключения внутрисхемного отладчика

РЕАЛИЗАЦИЯ

Весь аппаратный и программный инструментарий собрали, теперь самая пора начинать реализацию. Подключаем отладочную плату к отладчику, как показано на рис. 1, подключаем по USB и отладчик, и саму плату (при этом если установлены Alcohol 120% или Daemon Tools, то их нужно удалить, иначе отладчик в системе не определится), выставляем джамперы в соответствии с табл. 2. Теперь пойдём по простому пути. Запускаем STM32 ST-LINK Utility, открываем project/Obj/MP3_Play.hex и прошиваем. Все, теперь нашим самодельным айподом можно пользоваться.

Прошивка самодельного гаджета поддерживает работу с картами SD/SDHC до 32 Гб с файловой системой FAT16/FAT32, воспроизведение MP3/WMA/WAV/MID-файлов с частотой дискретизации 5–384 Кбит/с (файлы нужно складывать строго в папку /Music/ на карте), LRC-файлы (тексты песен с метками синхронизации, файлы складывать в папку /lrc/ на карте). Управление плеером — полностью и исключительно сенсорное. Есть три режима воспроизведения: Mode Rep — проигрывается один и тот же файл, Mode Cys — по порядку проигрываются циклично все файлы, Mode Rnd — проигрывание в случайном порядке всех файлов на карте.

КАСТОМИЗАЦИЯ

А теперь самое интересное — поговорим о том, как внутри все устроено и как подпиливать этот гаджет под себя.

Для микроконтроллеров с ядром Cortex-M3 (M0) базовой является библиотека CMSIS компании ARM — общая для всех микроконтроллеров всех производителей с данным ядром, стандарт взаимодействия ПО с ядром. Каждый производитель добавляет к ядру свой набор периферии и задействует те или иные возможности ядра, предоставляя библиотеку — надстройку над CMSIS. У STM32 такая библиотека именуется Standard Peripheral Library (SPL, библиотека драйверов периферийных устройств, в проекте нашего гаджета она в папке /source/Library). Если CMSIS компании ARM реализует программный интерфейс к самому ядру Cortex, то SPL компании ST реализует интерфейс к периферии микроконтроллера, лежащей за пределами ядра Cortex. CMSIS отдельно скачивать нет необходимости — она уже входит в комплект поставки SPL (bit.ly/NzzUu8).

В качестве графической библиотеки для рисования объектов на экране и работы с сенсорной панелью экрана выбрана адаптированная для STM32 библиотека Microchip Graphics Library (входит в состав бесплатного пакета Microchip Application Libraries, доступного по адресу: bit.ly/P6YU0f). Она содержит драйверы большинства распространенных контроллеров LCD-экранов, элементы взаимодействия с пользователем (кнопки, слайдеры, чекбоксы и так далее), то есть предоставляет интерфейс твоему приложению для работы с LCD-экраном и его сенсорной панелью. В проекте она лежит в /source/code/GUI/. К слову говоря, у производителя этих микроконтроллеров — компании ST есть своя собственная бесплатная графическая библиотека для микроконтроллеров STM32 — STM32 Embedded GUI Library, но она значительно менее распространена, и ее использование куда хуже разжевано в интернете, чем у Microchip Graphics Library.

Отдельно стоит упомянуть про одну из утилит, входящих в комплект этой библиотеки, — Graphic Resource Converter. Утилита позволяет конвертировать изображения в форматах BMP, JPEG, шрифты (как в виде отдельных файлов, так и установленных в систему, в том числе True Type) и бинарные файлы в HEX-формат (на выходе получаем Си-файл, который можно использовать в своем проекте). Именно таким образом в наш проект загружаются нестандартные иконки, кнопки и шрифты, в том числе кириллические. Шрифты, сгенерированные этой утилитой, хранятся в нашем проекте в файле source/Fonts.c, а картинки и иконки — в source/PicturesC32.c (кнопки «вперед», «назад», «стоп», «проигрывание» и иконка с изображением динамика).

РАБОТА СО СРЕДОЙ KEIL

Среды могут быть разными, мы будем рассматривать далее на примере Keil. После того как установили Keil, открываем им файл project/MP3_Play.proj, далее уже в самом Keil открываем «code → main.c» — это и есть «точка входа» программы MP3-плеера. Чтобы посмотреть, где описана та или иная функция, в исходнике кликаем правой кнопкой мыши по названию функции и в появившемся меню выбираем «Go To Definition Of '[название_функции]'». Чтобы посмотреть, как твоя программа работает на реальном устройстве, нужно проверить, что отладчик и плата MP3-плеера соединены друг с другом так, как показано на рис. 1, далее подключить по USB саму плату MP3-плеера (Hy Fireball) и плату отладчика (STM32 DISCOVERY) к компьютеру — при этом в системе отладчик должен определиться как STMicroelectronics STLink dongle. Плату MP3-плеера тоже необходимо подключать по USB, так как питания отладчика недостаточно для платы плеера. Для начала отладки выбираем «Debug → Start/Stop debug session» — откроется окно, как на рис. 2, и при этом в микроконтроллер загрузится уже собранная ранее прошивка MP3-плеера. Здесь можно отметить переменные твоей программы, изменение значений которых в микроконтроллере ты сможешь видеть на лету, — для этого в исходнике кликаем правой кнопкой мыши по названию нужной переменной и выбираем

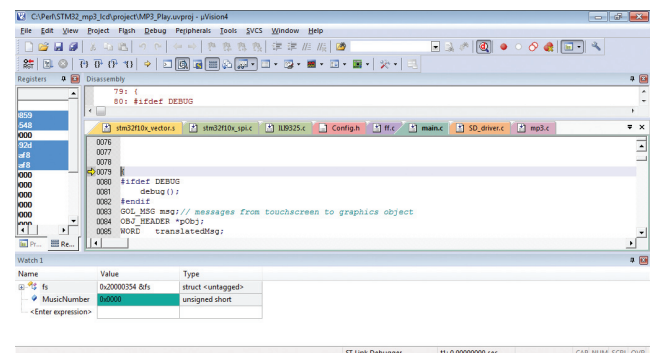


Рис. 2. Keil в режиме отладки

ем «Add 'название_переменной' to... -> Watch 1» — в дальнейшем значение указанной переменной можно будет увидеть в окне «View -> Watch Windows -> Watch 1».

Сбросить микроконтроллер можно командой «Debug -> Reset CPU». Чтобы начать выполнение программы в твоём девайсе, выбираем «Debug -> Run». В процессе отладки бывает необходимо приостановить работу программы на нужных участках кода (чтобы посмотреть значения переменных в этот момент, например). Для этого существуют «точки останова» — Breakpoints. Для этого два раза кликаем по номеру строки в файле исходника — номер помечается красным прямоугольником, указывающим, что при попадании на эту строку выполнение программы будет приостановлено.

Также довольно часто для отладки используют ком-порт. В нужных местах программа ставится обычный вызов printf(), который перенаправляет вывод в порт USART микроконтроллера. Такая техника используется и в нашем MP3-плеере. Подключаем USART1 к ком-порту компьютера, открываем терминалку и смотрим выводимые программой отладочные данные.

ЛОГИКА РАБОТЫ MP3-ПЛЕЕРА: ГРАФИЧЕСКИЙ ИНТЕРФЕЙС

Начнем наконец вникать в работу MP3-плеера «изнутри». Для начала — работа с графической библиотекой (рис. 3). Вначале необходимо правильно настроить ноги микроконтроллера для работы с экраном и включить на этих ногах тактирование — за это отвечает LCD_Configuration(). Отдельно настраиваются ноги, к которым подключен сенсорный экран, — TP_Init() (на экране стоит четырехпроводная резистивная сенсорная панель, которая подключена к микросхеме АЦП AD7843 на борту экрана, а уже та, в свою очередь, по интерфейсу SPI подключена к STM32). После этого нам необходимо проинициализировать библиотеку Microchip Graphics Library, вызвав GOLInit(), далее обстрелять байтами экран — задать ему настройки для корректной работы — LCD_Initialize(). Теперь графическая система нашего плеера готова к работе, но рисовать графический интерфейс на экране еще рано — при первом включении нужно откалибровать экран (предлагаем пользователю три раза нажать по разным углам экрана), для этого и вызываем TouchPanel_Calibrate(). Только после этого самая пора начинать рисовать интерфейс на экране вызовом встроенной функции библиотеки GOLDraw() — тем самым мы вызовем определенную нами GOLDrawCallback(). При первом вызове она вызовет CreateMp3Gui(), где перечисляем используемые цветовые схемы и определяем выводимые на экран графические объекты (кнопки, слайдеры и прочее), а при повторных вызовах будет менять выведенные на экран данные (например, время проигрывания файла). Заполняем структуру с координатами точки нажатия на сенсорном экране при помощи TouchGetMsg() и обрабатываем ее средствами графической библиотеки путем вызова GOLMsg() — эта функция определит, на какой объект пользователь

SWD (CN2)		JTAG (CN13)	
1	VDD_TARGET	1	3.3V power
2	SWCLK	4	GND
3	GND	7	PA13 (SWDIO)
4	SWDIO	9	PA14 (SWCLK)
5	NRST	13	PB3 (SWO)
6	SWO	15	RESET# (NRST)

Таблица 1. Разъемы для подключения внутрисхемного отладчика

Jumper	State
J5	2-3
JP9	1-2
JP5	1-2
JP3 (SD_CD)	Closed
JP1 (RST)	Open
JP2 (ISP)	Open
JP4 (BOOT1)	2-3
JP7 (BOOT0)	2-3
JP6 (VBAT)	1-2

Таблица 2. Схема установки джамперов на плате

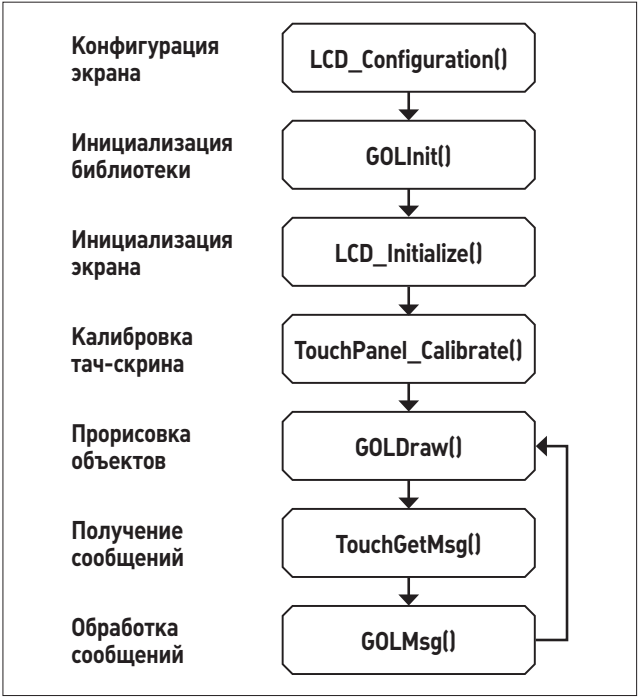


Рис. 3. Схема работы с графической библиотекой

нажал, и при следующем вызове GOLDraw() наша программа обработает это нажатие и внесет соответствующие изменения в вывод на экране. Таким образом, мы циклически бегем по цепочке GOLDraw() -> TouchGetMsg() -> GOLMsg().

ЛОГИКА РАБОТЫ MP3-ПЛЕЕРА: ЧТЕНИЕ И ВОСПРОИЗВЕДЕНИЕ МУЗЫКАЛЬНЫХ ФАЙЛОВ

За работу с папками и файлами на SD/SDHC-карте отвечает бесплатная китайская библиотека Чана — FatFs Module (bit.ly/NXY833), реализующая функции для работы с файловыми системами FAT16/FAT32 на микроконтроллерах ARM, AVR, PIC24 и многих других. Она не заточена под какой-то определенный носитель (SD-карта, USB-флеш) и при наличии соответствующего драйвера может работать хоть с жестким диском, подключенным к микроконтроллеру.

Работа по воспроизведению музыкальных файлов, за исключением рассмотренного выше вывода на экран, выглядит следующим образом:

Инициализация:

1. SPI_Configuration — конфигурация выводов STM32 для SD-карты.
2. SD_Init() — инициализация SD-карты.
3. Vs1003_Init() — конфигурация выводов STM32 для Vs1003.
4. Mp3Reset(), Vs1003SoftReset() — сброс Vs1003.
5. LoadPatch() — инициализация Vs1003.
6. f_mount() — монтирование файловой системы SD-карты.
7. FiltrateMusic() — считывание информации по MP3/WMA/WAV/MID-файлам на карте.

Рабочий цикл (воспроизведение):

1. f_open() — открытие выбранного музыкального файла.
2. FindLrc() — поиск соответствующего lrc-файла для открытого ранее файла.
3. Vs1003_CMD_Write(SPI_VOL, volt) — установка выбранного уровня громкости.
4. Пишем в Vs1003 аудиоданные блоками по 512 байт. Аудиокодек каждый раз, когда готов принять от нас очередную порцию данных для воспроизведения, дергает линию DREQ.

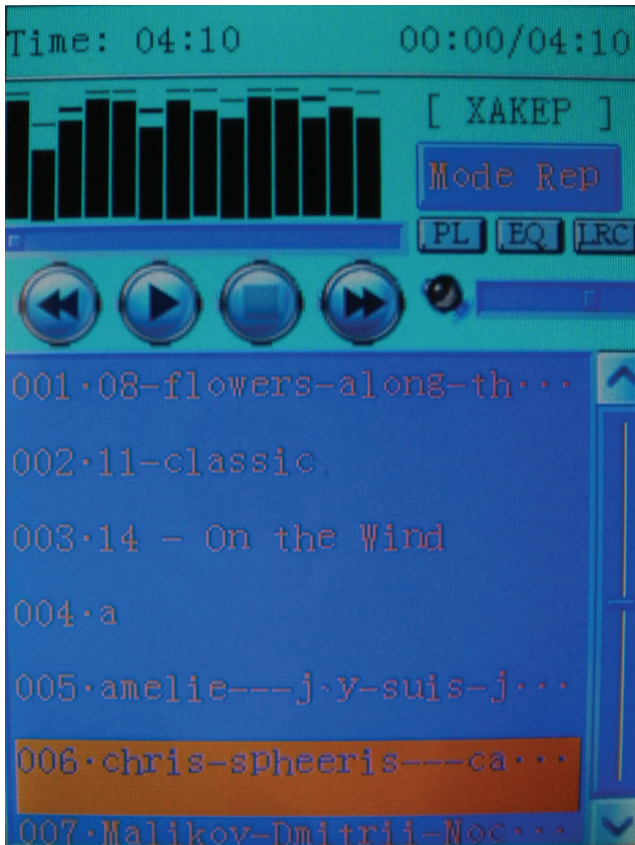


Рис. 4. Графический интерфейс MP3-плеера

ЧТО ДАЛЬШЕ?

На самом деле вовсе не обязательно использовать аппаратный MP3-декодер (Vs1003). Микроконтроллер STM32 вполне способен самостоятельно справиться с задачей декодирования MP3, и для этого производитель (ST) предоставляет бесплатную библиотеку для работы с MP3. Но по патентным соображениям данная библиотека не распространяется свободно, требуется подписание соглашения на ее использование — в основном по этой причине декодирование выполняется в нашем плеере отдельной микросхемой. Также существует вариант использования Helix MP3 Decoder (datatype.helixcommunity.org/Mp3dec) для программного декодирования MP3 силами STM32.

Во многих MP3-плеерах имеется встроенный FM-радиоприемник, который можно добавить и в наш плеер. Наиболее распространена для этих целей пуская и не новая, но зато дешевая (2 доллара) микросхема TEA5767, которая общается с микроконтроллером через интерфейс I2C и не требует внешней антенны. В интернете можно найти много примеров работы с ней плюс ее не нужно искать в магазинах — велика вероятность, что, если разберешь старый ненужный плеер с FM-радио, увидишь эту микросхему. Также можно купить отладочную плату на eBay. Встроенной поддержки RDS нет — предлагается использовать для этого отдельный RDS-декодер SAA6588, либо можно перейти на использование более новой, но куда более дорогой микросхемы FM-радио Si4735 — там поддержка RDS уже встроена.

Наигравшись с воспроизведением MP3 и прослушиванием радио, ты можешь захотеть добавить возможность просмотра фильмов в MPEG4. Просмотр видео можно реализовать на STM32, но в ограниченном виде (вряд ли сумеешь выжать более 20 fps 16-bit QVGA видео на Cortex M3 либо 30 fps QVGA MotionJPEG на Cortex M4), поэтому тут нужно будет переходить на использование

старших Cortex'ов. Для этого можно присмотреться к отладочной плате FriendlyARM (например, FriendlyARM Mini2440 — 85 долларов на eBay), на которую ставятся микроконтроллеры с ядром ARM8/9/11. А это уже возможность запуска Linux/WinCE/Android на самодельном девайсе (в случае с STM32 линукс не поставить, только его обрезанный вариант ucLinux). Таким образом можно плавно перейти от создания своего Apple iPod к разработке самодельного Apple iPad.


Но в большинстве случаев использование старших кортексов — это стрельба из пушки по воробьям, плюс с ними на порядок сложнее работать, чем с Cortex M, поэтому сейчас наиболее распространены STM32 при решении бытовых задач. И уж точно не стоит начинать изучение микроконтроллеров с FriendlyARM.

В заключение не могу не упомянуть про один очень полезный инструмент для отладки микроконтроллерных (и не только) устройств. Допустим, у нас есть ненужный рабочий промышленный MP3-плеер с микросхемой FM-радио на борту, эту микросхему хочется прикрутить к своему девайсу и из даташита не совсем понятно, как с ней работать. Тут на помощь приходит логический анализатор, которым мы можем подружиться к исследуемому устройству и посмотреть, как производится обмен данными с микросхемой — что, например, нужно сделать, чтобы увеличить громкость. Также бывает необходимость в своем устройстве посмотреть, «что же там происходит на уровне сигналов», — внутрисхемная отладка далеко не всегда позволяет решить все проблемы.

Наиболее популярны логические анализаторы (которые также умеют работать и как осциллографы) USBee AX/DX и Saleae — подключаются к компьютеру через USB-интерфейс, на выходе — щупы для подключения к отлаживаемому устройству. На eBay/Alibaba/DealExtreme большое разнообразие клонов, которые умеют работать с софтом как USBee, так и Saleae, по цене от 20 долларов. Также у них есть поддержка большинства популярных протоколов — I2C, SPI, CAN, USB, RS232, RS485 и множества других.

ВМЕСТО ПОСЛЕСЛОВИЯ

Вот и все — плеер готов. По ссылке files.mail.ru/0F2SSL выложен полный проект MP3-плеера со всеми исходниками, можешь ковырять его и совершенствовать.

Даже если ты никогда раньше не имел дело с микроконтроллерами, этот самодельный MP3-плеер будет полезен и наверняка даст стимул и дальше осваивать эту область, ведь куда приятнее начинать изучение не с банального моргания светодиодом, а с чего-то более полезного и функционального. А куда развиваться дальше — решать тебе: можешь добавить поддержку OBD-II и сделать бортовой компьютер для машины медиацентром или целую систему домашней автоматизации. Не бойся и твори! 

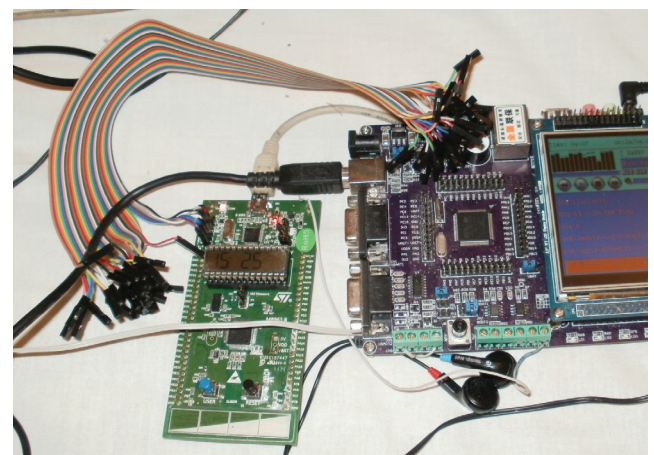


Рис. 5. MP3-плеер в сборе с отладчиком