mrp-docs-ru Documentation

Выпуск 0.1

D-Free-J

30 December 2013

Оглавление

Содержание:

Общие сведения о платформе MRP (MiniJ)

1.1 О платформе

Платформа MRP (MiniJ) - это набор API, поддерживаемых на ряде сотовых телефонов, произведённых в Китае.

Цель создания платформы - замена Java на мобильном устройстве.

Для распространения программ используются файлы типа MRP (см. Формат MRP)

Глава 2

Форматы и структура файлов

Стандартные типы используемых файлов:

2.1 Формат BMP

Файл, предназначенный для хранения изображений.

Расширение - bmp.

Хотя расширение и совпадает с расширением BMP файл хранит только сам растр, 16 бит на точку в формате RRRRRGGG GGGBBBBB (5 бит красный, 6 - зелёный, 5 - синий), начиная с верхней сроки. Сведения о ширине и высоте изображения не хранятся внутри.

Порядок следования бит может отличаться для разных типов аппаратных платформ.

При компиляции стандартный компилятор автоматически выполняет конвертацию в этот формат, с учётом аппаратной платформы.

2.2 Формат EXT

Файл, предназначенный для хранения скомпилированного кода, специфичного для текущей аппаратной платформы.

Расширение - ext.

2.3 Формат MR

Файл, предназначенный для хранения байт-кода виртуальной машины.

Расширение - mr.

2.3.1 Типы данных

Целые числа хранятся в Little Endian (LE).

Строки в файле хранятся в формате:

- длина строка (uint32, целое без знака 4 байта)
- строка (последовательность байт)
- нулевой байт, входит в длину строки.

Строка не несёт в себе информации о кодировке. За выбор нужной кодировки отвечает работающая со строкой программа.

Каждый массив предваряется его размером - целым числом без знака (uint32). После идут его элементы.

2.3.2 Структура файла

Структура файла представлена в таблице 1.

Таблица 1

Смещение	Тип
0x00	MAGIC константа
0x04	Версия
0x06	Главная функция

2.3.3 MAGIC

Константа для идентификации типа файла. Имеет значение 0x1B4D5250 (“\x1bMRP”).

2.3.4 Версия

Два байта подряд 0x8001. Другие значения не встречались.

2.3.5 Функции

Исполнение байт-кода начинается с главной функции. Имя главной функции значения не имеет и обычно совпадает с именем компилируемого файла. Главная функция имеет точно такой же формат что и вложенные в неё.

Функция состоит из следующих секций:

- строка - имя функции, для главной функции - имя файла. Может быть пустым.
- uint32 - номер строки на которой функция объявлена
- uint8 - количество переменных из внешних функций
- uint8 - количество аргументов функции
- uint8 - не ноль если количество аргументов может изменяться
- uint8 - максимальная глубина использования стека

- Массив номеров строк на которых были операторы этой функции (тип - uint32)
- Массив локальных переменных
  - строка - имя переменной
  - uint32 - начало области действия
  - uint32 - конец области действия
- Массив используемых переменных из внешних функций (тип - строка)
- Массив констант
  - тип константы
    - 00 - nil, значения нет
    - 01 - bool, значение - 1 байт (Истина если не ноль)
    - 03 - int32, значение - 4 байта, целое со знаком
    - 04 - строка, значение - строка
  - значение
- Массив вложенных функций (тип - функция)
- Массив байт-кода (тип - uint32)

Байт-код

Младшие 6 бит используются как код оператора, остальные - аргументы.

2.4 Формат MRP

Файл, предназначенный для распространения программ платформы MRP. Также может использоваться как контейнер для хранения файлов. Файлы, содержащиеся внутри, обычно сжаты (формат gzip).

Расширение - mrp.

(см. Программы для работы с файлами MRP)

2.4.1 Структура файла

Структура файла представлена в таблице 1.

Таблица 1

Смещение	Тип
0x00	MAGIC константа
0x04	Заголовок
0xF0	Дополнительные данные в заголовке
...	Индексная таблица
...	Файловая таблица

2.4.2 MAGIC

Константа для идентификации типа файла. Имеет значение 0x4D525047 (“MRPG”).

2.4.3 Заголовок

Структура заголовка приведена в таблице 2.

Все строки фиксированной длины должны быть дополнены нулями до нужного размера. Кодировка символов “Китайский упрощённый GB-2312” (кроме поля file_name), определение кодировки выполняется непосредственно программой читающей эти строки, например, приложением отображающем список установленных программ (dsm_gm.mrp, mopo.mrp).

Все цифровые поля в Little Endian (LE), за исключением указанных.

Таблица 2

Смещение	Длина	Значение	Описание
0x04	4	headlen	Начало файловой таблицы
0x08	4	filelen	Длина файла
0x0C	4	index	Начало индексной таблицы
0x10	12	filename	Имя файла при компиляции (может не совпадать текущим именем файла), кодировка ascii
0x1C	24	appame	Строка имя программы
0x34	16	authcode	Код авторизации компилятора
0x44	4	appid	ID приложения
0x48	4	appver	Версия приложения
0x4C	4	flags	Флаги
0x50	4	formatver	Версия формата
0x54	4	crc32	Контрольная сумма CRC32. Вычисляется от всего файла, за исключением самого поля, вместо него 4 нулевых байта.
0x58	40	vendor	Строка имя производителя (или автора)
0x80	64	description	Строка описание
0xC0	4	appid	ID приложения (в BE)
0xC4	4	appver	Версия приложения (в BE)
0xC8	4	UNKNOWN	Here dragons...
0xCC	2	scr_width	Ширина экрана (или 0)
0xCE	2	scr_height	Высота экрана (или 0)
0xD0	1	platform	Тип аппаратной платформы
0xD1	31	padding	31 нулевой байт (выравнивание для совместимости)
0xF0	...		Дополнительные данные

Поле headlen

Поле определяет длину заголовка и, соответственно, начало файловой таблицы. Если значение не более 232 байт - то файл использует старый формат.

Старый формат отличается тем, что не имеет индексной таблицы. Из-за этого для поиска файла с конкретным именем необходимо просматривать файловую таблицу полностью.

Кроме того после окончания полей заголовка могут размещены дополнительные данные, в этом случае начало индексной таблицы сдвигается (поле index_ref).

Так как размер указан без учёта константы MAGIC то начало файловой таблицы будет определяться как headlen + 4.

Поле authcode

Authorization code, no-brain task.

Поле flags

Биты:

0 - visible (видимость в списке программ)
1 .. 2 - cpu (тип процессора)
3 .. - shell

Поле formatver

Format version: 0x2710 (10000) или 0x2712 (10002)

Другие значения пока не встречались.

Поле platform

0 - не указано
1 - MTK или MSTAR
2 - SPREADTRUM

Тип аппаратной платформы (см. Аппаратные платформы) указывается для справки, при запуске программы соответствие не всегда проверяется. Бинарные компоненты (.ext) должны быть скомпилированы для конкретной аппаратной платформы.

Дополнительные данные в заголовке

Некоторые программы используют данные размещённые после последнего поля в заголовке и до начала индексной таблицы. В этом случае начало индексной таблицы также сдвигается (см. поле index).

2.4.4 Индексная таблица

Индексная таблица содержит записи обо всех файлах с указанием их расположения в файле.

Структура записи указана в таблице 3.

Таблица 3

Поле	Размер	Описание
nlen	4	Длина имени файла в байтах, с учётом нулевого байта
name	nlen	Имя файла в архиве
start	4	Позиция данных в файле
len	4	Длина файла, байт
pad	4	Дополнение (padding)

Записи повторяются до начала файловой таблицы.

2.4.5 Файловая таблица

Файловая таблица содержит непосредственно данные файлов.

Поле	Размер	Описание
nlen	4	Длина имени файла в байтах, с учётом нулевого байта
name	nlen	Имя файла в архиве
len	4	Длина файла, байт
data	len	Данные. На эту позицию указывают записи в индексной таблице

Записи повторяются до конца файла. Количество записей должно совпадать с количеством записей в индексной таблице.

Так как поля с именем файла и длиной из индексной таблицы повторяются в файловой они могут быть повреждены или даже отсутствовать. Корректность работы в этом случае зависит только от конкретной реализации платформы.

Типы файлов, используемые приложениями:

2.5 Формат SLG

REing: MaReW

Файл, предназначенный для хранения изображений.

Расширение - slg.

2.5.1 Структура файла

Существует два типа файлов: обычные и упакованные.

Структура обычного файла представлена в таблице 1. Целые числа хранятся в Big Endian (BE).

Таблица 1

Смещение	Тип
0x00	MAGIC константа
0x04	Ширина (uint16)
0x06	Высота (uint16)
0x08	Размер данных (uint32)
0x0c	Версия формата (uint32)
0x10	Данные

Структура сжатого файла представлена в таблице 2. Данные упакованы в формате gzip. Целые числа хранятся в Little Endian (LE).

Таблица 2

Смещение	Тип
0x00	MAGIC константа
0x04	Ширина (uint16)
0x06	Высота (uint16)
0x08	Размер данных (uint32)
0x0c	Данные

2.5.2 MAGIC

Константа для идентификации типа файла.

- 0x736c6730 (“slg0”) - без сжатия
- 0x0000efbb (“\x00\x00\xef\xbb”) - с сжатием

2.5.3 Размер данных

Для несжатых файлов указывается размер всего поля данных до конца файла.

Для сжатых - размер после упаковки.

2.5.4 Версия формата

Это поле есть только в несжатых файлах. Значение должно быть равно 6 (другие значения не встречались).

2.5.5 Данные

Вне зависимости от того используется ли сжатие структура данных имеет одинаковый формат:

- Растр
- Альфа-канал

2.5.6 Растр

Размер растра = ширина * высота * 2.

Растр хранится в виде 16 бит на точку в формате RRRRRGGG GGGBBBBB (5 бит красный, 6 - зелёный, 5 - синий), начиная с верхней строки.

2.5.7 Альфа-канал (прозрачность)

Размер этого поля равен размеру всего поля данных без растра.

Каждый байт имеет формат TTVVVVVV и определяет прозрачность каждого пикселя, начиная с верхней строки. T - два бита (0 - 3), V - 6 бит (0 - 63).

Поле может T (тип) иметь значение:

- 0 - окончание строки (V тоже равен 0). Должен быть после каждой строки.

- 1 - прозрачный сегмент, количество прозрачных пикселей равно значению V.
- 2 - полупрозрачный пиксель, значение V находится в диапазоне от 0 (полностью прозрачный) до 31 (непрозрачный).
- 3 - непрозрачный сегмент, количество непрозрачных пикселей равно значению V.

При типе 2 один бит (6, старший бит) поля V остаётся неиспользованным.

2.6 Формат BM

Файл, предназначенный для хранения изображений.

Расширение - bm.

2.6.1 Структура файла

Структура файла представлена в таблице 1. Целые числа хранятся в Little Endian (LE).

Таблица 1

Смещение	Тип
0x00	MAGIC константа
0x04	Ширина (uint16)
0x06	Высота (uint16)
0x08	Данные

2.6.2 MAGIC

Константа для идентификации типа файла 0x534b42d4 (“SKBM”).

2.6.3 Данные

Размер поля = ширина * высота * 2.

Растр хранится в виде 16 бит на точку в формате RRRRRGGG GGGBBBBB (5 бит красный, 6 - зелёный, 5 - синий), начиная с верхней строки.

Как это сделано

Описание работы всех внутренних частей и взаимодействие между ними.

Примечание:

> Данный раздел опирается на исследование работы конкретных реализаций платформы MRP. Возможны нюансы.

3.1 Запуск приложения

3.1.1 Общие сведения

Каждое запускаемое приложение упаковано в контейнер (см. Формат MRP).

Запуск начинается с файла с именем start.mr (см. Формат MR).

Байт-код приложения загружается в память до окончания работы приложения. В главном файле определены точки входа для реакции на различные события (init, resume, dealevent и т. п.).

Приложения обычно запускаются загрузчиком. Загрузчик может быть встроен в прошивку или находится в файловой системе телефона (в этом случае его можно подменить, на любой работающий MRP файл).

Загрузчик обычно содержит только байт-код без бинарных компонент .ext (см. Формат EXT), но есть исключения.

3.1.2 Загрузчики

dsm_gm.mrp

Вариант загрузчика. Некоторые версии содержат бинарные компоненты (например, для запуска эмулятора NES). Может самостоятельно скачивать приложения.

mopo.mrp

Вариант загрузчика. Содержит большое количество бинарных компонент. Может самостоятельно скачивать приложения из интернета, в том числе требующие платную активацию.

3.2 Список использованных технологий

3.2.1 gzip

Формат хранения сжатых файлов, http://www.ietf.org/rfc/rfc1952.txt

Используется в файлах MRP (см. Формат MRP), SLG (Формат SLG)

3.2.2 zlib

Алгоритм zlib, http://www.zlib.net

Используется в файлах MRP (см. Формат MRP), SLG (Формат SLG)

3.3 Аппаратные платформы

Здесь приведён список аппаратных платформ. Термин платформа далее используется для обозначения аппаратной платформы, но не платформы MRP в целом.

Платформа - сочетание процессора, аппаратных особенностей, а не наличия клавиатуры, разрешения экрана и т. п.

В большинстве случает запуск бинарного кода для другой платформы приводит к перезагрузке аппарата.

3.3.1 MTK

Большинство аппаратов.

3.3.2 SPREADTRUM

Он же SPD.

Меньшая часть аппаратов

3.3.3 Win32

Используется при отладке приложения через официальный SDK (см. Официальный SDK). При запуске эмулятора бинарные компоненты компилируются в код для win32 в код самого приложения. Это связано с тем что пока нет эмулятора выполняющего код arm.

Средства разработки

Описание способов разработки приложений (см. также Программное обеспечение):

4.1 Официальный SDK

В интернете.

4.2 Примеры программ

Примеры программ

4.2.1 Только байт-код

Пример приложения, использующего только байт-код

```
def dealevent(code, p0, p1, p2)
  print("dealevent is called")

  str = string.format("code = %d, p0 = %d, p1 = %d", code, p0, p1)
  print(str)

  if p0 == 18 then
    Exit()
  end

  return 1
end


def suspend()
  print("suspend is called")
  return 1
end

def resume()
```

```
  print("resume is called")
  return 1
end

def init()
  print("init is called")
  return 1
end

def sampleapp()
  print("App is called")
  local sysinfo = GetSysInfo()

  ClearScreen(50,100,100)

  local w = sysinfo.scrw
  local h = sysinfo.scrh

  local hpos = 5
  local hdelta = 18
  DrawText("MR code test.", 5, hpos, 255, 255, 0)
  hpos = hpos + hdelta
  _drawLine(0, hpos - 1, w, hpos - 1, 255, 255, 0)

  local st = "Screen " .. w .. " x " .. h
  DrawText(st, 5, hpos, 255, 255, 0)
  hpos = hpos + hdelta

  local st = "Platform " .. sysinfo.hsman
  DrawText(st, 5, hpos, 255, 255, 0)
  hpos = hpos + hdelta

  local st = "IMEI " .. sysinfo.IMEI
  DrawText(st, 5, hpos, 255, 255, 0)
  hpos = hpos + hdelta * 2

  DrawText("Hello, world", 5, hpos, 255, 255, 255)
  hpos = hpos + hdelta
  DrawText(" sample for mrp-docs-ru :)", 5, hpos, 255, 255, 255)

  local st = "Exit"
  DrawText(st, w - 5 - _textWidth(st), h - 20, 255, 255, 0)


  _dispUp(0, 0, w, h)

end

sampleapp()
```

Компиляция файла

```
mrptool mr2mrp samplemronly.mr
```

Результат


MR code test.
Screen 240 x 320
Platform sdk
IMEI 1
Hello, world
sample for mrp-docs-ru :)
Exit

Программное обеспечение

Приложения (см. также Официальный SDK):

5.1 Программы для работы с файлами MRP

Программы для работы с файлами формата MRP (см. Формат MRP).

5.1.1 mrpextractor

Совместимость: win

Возможности:

- Распаковка MRP
- Упаковка (добавление файлов)
- Преобразование изображений в BMP (только 16 бит)
- Перевод заголовков (требуется подключение к интернет)

5.1.2 MRP_PRJ1

Совместимость: win

Возможности:

- Распаковка MRP
- Упаковка (добавление файлов)
- Преобразование изображений в BMP (любая губина цвета)

Глава 6

Примечания

Документация является компиляцией сведений из разных источников. Авторство указано по желанию самих авторов.

Все технологии являются собственностью их владельцев, если это доказано.

Список использованных технологий приведён в документе Список использованных технологий

Ссылки:

- Страница с исходниками https://github.com/d-free-j/mrp-docs-ru
- Если вы желаете дополнить информацию:
  - Присылайте pull request в формате reStructuredText на https://github.com/d-free-j/mrp-docs-ru/pulls
  - Присылайте txt/odt/doc файл