原 FFmpeg源代码简单分析：configure

2015年03月24日 10:22:53　阅读数：32197

==========

FFmpeg的库函数源代码分析文章列表：

【架构图】

FFmpeg 源代码结构图 - 解码

FFmpeg 源代码结构图 - 编码

【通用】

FFmpeg 源代码简单分析： av_register_all()

FFmpeg 源代码简单分析： avcodec_register_all()

FFmpeg 源代码简单分析：内存的分配和释放（ av_malloc()、 av_free() 等）

FFmpeg 源代码简单分析：常见结构体的初始化和销毁（ AVFormatContext， AVFrame 等）

FFmpeg 源代码简单分析： avio_open2()

FFmpeg 源代码简单分析： av_find_decoder() 和 av_find_encoder()

FFmpeg 源代码简单分析： avcodec_open2()

FFmpeg 源代码简单分析： avcodec_close()

【解码】

图解 FFMPEG 打开媒体的函数 avformat_open_input

FFmpeg 源代码简单分析： avformat_open_input()

FFmpeg 源代码简单分析： avformat_find_stream_info()

FFmpeg 源代码简单分析： av_read_frame()

FFmpeg 源代码简单分析： avcodec_decode_video2()

FFmpeg 源代码简单分析： avformat_close_input()

【编码】

FFmpeg 源代码简单分析： avformat_alloc_output_context2()

FFmpeg 源代码简单分析： avformat_write_header()

FFmpeg 源代码简单分析： avcodec_encode_video()

FFmpeg 源代码简单分析： av_write_frame()

FFmpeg 源代码简单分析： av_write_trailer()

【其它】

FFmpeg 源代码简单分析：日志输出系统（ av_log() 等）

FFmpeg 源代码简单分析：结构体成员管理系统 -AVClass

FFmpeg 源代码简单分析：结构体成员管理系统 -AVOption

FFmpeg 源代码简单分析： libswscale 的 sws_getContext()

FFmpeg 源代码简单分析： libswscale 的 sws_scale()

FFmpeg 源代码简单分析： libavdevice 的 avdevice_register_all()

FFmpeg 源代码简单分析： libavdevice 的 gdigrab

【脚本】

FFmpeg 源代码简单分析： makefile

FFmpeg 源代码简单分析： configure

【H.264】

FFmpeg 的 H.264 解码器源代码简单分析：概述

==========

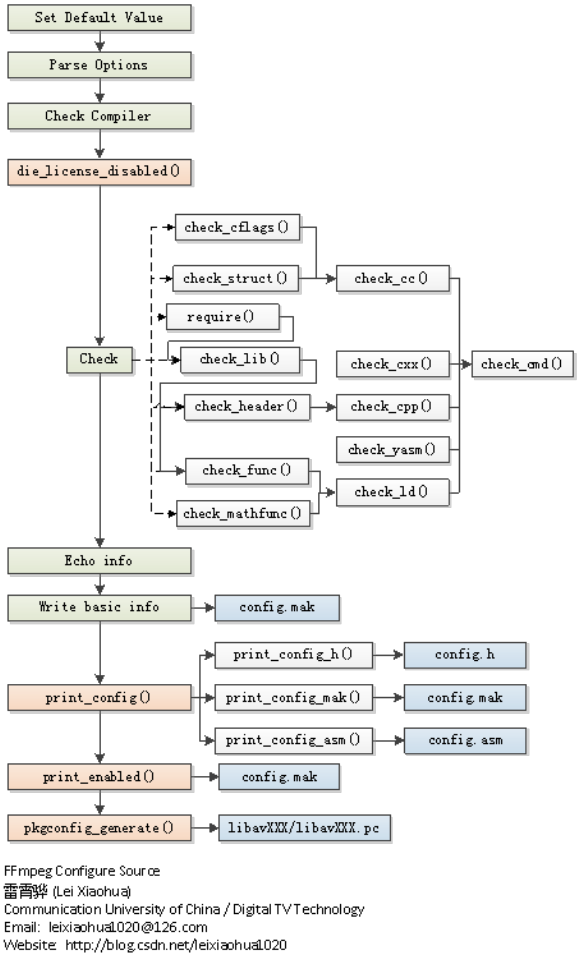
本文记录FFmpeg的Configure脚本的源代码。Configure一方面用于检测FFmpeg的编译环境，另一方面根据用户配置的选项生成config.mak，config.h文件（可能还有config.asm），提供给Makefile使用。由于FFmpeg的configure脚本很复杂（一个4000-5000行的Shell脚本），难以逐行细致的分析，因此本文简单梳理一下它的结构。

PS1：Configure的日志位于config.log文件中。查看该文件有助于分析Configure的过程。

PS2：使用“sh -x script_name.sh”可以调试Shell脚本。

Configure文件的整体流程

Configure文件的整体流程如下所示。



Configure的整体流程可以分成以下几步：

Set Default Value：设置各个变量默认值；

Parse Options：解析输入的选项；

Check Compiler：检查编译器；

die_license_disabled()：检查GPL等协议的设置情况；

Check：检查编译环境（数学函数，第三方类库等）；

Echo info：控制台上打印配置信息；

Write basic info：向config.mak中写入一些基本信息；

print_config()：向config.h、config.mak、config.asm中写入所有配置信息；

print_enabled()：向config.mak写入所有enabled的组件信息；

pkgconfig_generate()：向libavXXX/libavXXX.pc中写入pkgconfig信息（XXX代表avcodec，avformat等）；

下文简单梳理一下这些步骤。

Set Default Value

Set Default Value部分设置一些Configure的默认值。例如下面的代码。

```python
# 默认参数 default parameters
# 日志
logfile="config.log"

# 安装路径 installation paths
prefix_default="/usr/local"
bindir_default='${prefix}/bin'
datadir_default='${prefix}/share/ffmpeg'
incdir_default='${prefix}/include'
libdir_default='${prefix}/lib'
mandir_default='${prefix}/share/man'
shlibdir_default="$libdir_default"
postproc_version_default="current"

# 工具链 toolchain
ar_default="ar"
cc_default="gcc"
cxx_default="g++"
cc_version=\"unknown\"
host_cc_default="gcc"
install="install"
ln_s="ln -sf"
nm_default="nm"
objformat="elf"
pkg_config_default=pkg-config
ranlib="ranlib"
strip_default="strip"
yasmexe_default="yasm"

nm_opts='-g'
nogas=":"

# 机器 machine
arch_default=$(uname -m)
cpu="generic"

# 操作系统 OS
target_os_default=$(tolower $(uname -s))
host_os=$target_os_default

# alternative libpostproc version
ALT_PP_VER_MAJOR=51
ALT_PP_VER_MINOR=2
ALT_PP_VER_MICRO=101
ALT_PP_VER=$ALT_PP_VER_MAJOR.$ALT_PP_VER_MINOR.$ALT_PP_VER_MICRO

# 选项 configurable options
# PROGRAM_LIST内容是 ffplay ffprobe ffserver ffmpeg
enable $PROGRAM_LIST

enable avcodec
enable avdevice
enable avfilter
enable avformat
enable avutil
enable postproc
enable stripping
enable swresample
enable swscale

enable asm
enable debug
enable doc
enable fastdiv
enable network
enable optimizations
enable safe_bitstream_reader
enable static
enable swscale_alpha

# 编译选项 build settings
SHFLAGS='-shared -Wl,-soname,$$(@F)'
```

```
SHFLAGS='-shared -Wl,-soname,$$(@F)'
FFSERVERLDFLAGS=-Wl,-E
# 前缀后缀
LIBPREF="lib"
LIBSUF=".a"
FULLNAME='$(NAME)$(BUILDSUF)'
# 名称
LIBNAME='$(LIBPREF)$(FULLNAME)$(LIBSUF)'
# 动态库前缀后缀
SLIBPREF="lib"
SLIBSUF=".so"
# 名称
SLIBNAME='$(SLIBPREF)$(FULLNAME)$(SLIBSUF)'
SLIBNAME_WITH_VERSION='$(SLIBNAME).$(LIBVERSION)'
SLIBNAME_WITH_MAJOR='$(SLIBNAME).$(LIBMAJOR)'
LIB_INSTALL_EXTRA_CMD='$$(RANLIB) "$(LIBDIR)/$(LIBNAME)"'
SLIB_INSTALL_NAME='$(SLIBNAME_WITH_VERSION)'
SLIB_INSTALL_LINKS='$(SLIBNAME_WITH_MAJOR) $(SLIBNAME)'

AS_O='-o $@'
CC_O='-o $@'
CXX_O='-o $@'

host_cflags='-D_ISOC99_SOURCE -O3 -g'
host_libs='-lm'

target_path='$(CURDIR)'
```

需要注意的是，“enable avcodec”，“enable avformat”，“enable avutil”等中的enable()本身是一个函数。enable()的定义如下。

```python
#把所有输入参数的值设置为"yes"
enable(){
    set_all yes $*
}
```

可以看出enable()调用了set_all()函数。并且将第1个参数设置为“yes”，并且将调用enable()时候的参数传递给set_all()。set_all()函数的定义如下所示。

```python
#第一个参数为值，后面的参数为变量
set_all(){
    value=$1
    shift
    for var in $*; do
        eval $var=$value
    done
}
```

可以看出set_all()将传入的参数全部进行赋值。特定于enable()函数来说，就是将所有的输入变量赋值为“yes”。由此可见，“enable avcodec”实际上相当于执行了：

```python
avcodec=”yes”
```

Parse Options

Parse Options部分用于解析Configure的附加参数。该部分的代码如下所示。

```python
#注意：opt不是参数列表（实际上也没有看见opt变量的定义）
#原因是处在for循环中，当你没有为in指定列表时，for会默认取命令行参数列表。
#因此“opt”这个名字实际上是可以随便取的
for opt do
# "#"用于去除特定字符前面的字符串
# optval内容为opt去掉"="以及其前面字符串之后的内容
    optval="${opt#*=}"
    case "$opt" in
    # 不同的选项
    --extra-ldflags=*) add_ldflags $optval
    ;;
    --extra-libs=*) add_extralibs $optval
    ;;
    --disable-devices) disable $INDEV_LIST $OUTDEV_LIST
    ;;
    --enable-debug=*) debuglevel="$optval"
    ;;
    --disable-everything)
    map 'eval unset \${$(toupper ${v%s})_LIST}' $COMPONENT_LIST
    ;;
    --enable-*=*|--disable-*=*)
    eval $(echo "${opt%%=*}" | sed 's/--/action=/;s/-/ thing=/')
    is_in "${thing}s" $COMPONENT_LIST || die_unknown "$opt"
    eval list=\$$(toupper $thing)_LIST
    name=$(echo "${optval}" | sed "s/,/_${thing}|/g")_${thing}
    $action $(filter "$name" $list)
    ;;
    --enable-?*|--disable-?*)
    eval $(echo "$opt" | sed 's/--/action=/;s/-/ option=/;s/-/_/g')
    if is_in $option $COMPONENT_LIST; then
        test $action = disable && action=unset
        eval $action \$$(toupper ${option%s})_LIST
    elif is_in $option $CMDLINE_SELECT; then
        $action $option
    else
        die_unknown $opt
    fi
    ;;
    --list-*)
        NAME="${opt#--list-}"
        is_in $NAME $COMPONENT_LIST || die_unknown $opt
        NAME=${NAME%s}
        eval show_list $NAME \$$(toupper $NAME)_LIST
    ;;
    --help|-h) show_help
    ;;
    *)
    #% 就是从右边开始删除符合条件的字符串（符合条件的最短字符串）
    #%%是删除符合条件的最长的字符串

    #删除“=”右边的内容
    optname="${opt%%=*}"
    #删除左边的“--”
    optname="${optname#--}"
    optname=$(echo "$optname" | sed 's/-/_/g')
    #看看是否在opt列表中，不在的话就会返回错误
    if is_in $optname $CMDLINE_SET; then
        eval $optname='$optval'
    elif is_in $optname $CMDLINE_APPEND; then
        append $optname "$optval"
    else
         die_unknown $opt
    fi
    ;;
    esac
done
```

在这里需要注意，取出opt的值一般都是“--extra-ldflags=XXX”的形式，通过“${opt#*=}”截取获得“=”号后面的内容作为optval，对于“--extra-ldflags=XXX”来说，optval取值为“XXX”。
然后根据opt种类的不同，以及optval取值的不同，分别作不同的处理。

Check Compiler

Check Compiler用于检查编译器。这部分代码还没有细看，暂时不做分析。

die_license_disabled()

die_license_disabled()用于检查是否指定了特定了License。像libx264、libfaac这些第三方类库，都需要指定特定的License才可以使用（例如libfaac必须指定nonfree）。开启这些第三方类库后如果没有指定License，Configure会立刻退出。这部分代码如下所示。

```python
#检查License
#GPL
die_license_disabled gpl libcdio
die_license_disabled gpl libx264
die_license_disabled gpl libxavs
die_license_disabled gpl libxvid
die_license_disabled gpl x11grab
#nonfree
die_license_disabled nonfree libaacplus
die_license_disabled nonfree libfaac
die_license_disabled nonfree openssl
#Version3
die_license_disabled version3 libopencore_amrnb
die_license_disabled version3 libopencore_amrwb
die_license_disabled version3 libvo_aacenc
die_license_disabled version3 libvo_amrwbenc
```

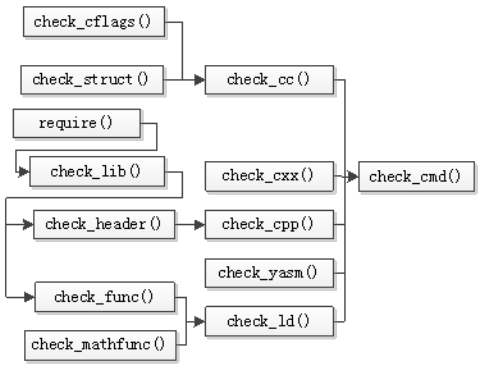
其中涉及到一个函数die_license_disabled()，它的定义如下所示。

```python
#不符合License则立刻结束
die_license_disabled() {
    enabled $1 || { enabled $2 && die "$2 is $1 and --enable-$1 is not specified."; }
}
```

从定义可以看出，die_license_disabled()首先会看第1个参数（对应“gpl”，“nonfree”）对应的组件是否enable，如果已经enable，则正常运行完函数；如果没有enable，则会检查第2个参数（对应“libx264”，“libfaac”）是否enable，如果第2个参数enable了，就会报错退出。

Check

Check部分是Configure中最重要的部分。该部分用于检查编译环境（例如数学函数，第三方类库等）。这一部分涉及到很多的函数。包括check_cflags()、check_struct()、require()、check_lib()、check_header()、check_func()、check_mathfunc()等等。这些函数之间的调用关系如下图所示。



下面简单举例下面几个函数：

check_func()：用于检查函数。

check_header()：用于检查头文件。

check_func_headers()：用于同时检查头文件和函数。

check_mathfunc()：用于检查数学类函数。

require()：检查第三方类库。

check_cflags ()：用于检查编译器的cflags标志参数。

下面详细看看这几个函数。

check_func()

check_func()用于检查函数。它的输入参数一个函数名。Configure中与check_func()有关的代码如下所示。

```python
check_func  isatty
check_func  localtime_r
check_func  ${malloc_prefix}memalign            && enable memalign
check_func  mkstemp
check_func  mmap
check_func  ${malloc_prefix}posix_memalign      && enable posix_memalign
check_func  setrlimit
check_func  strerror_r
check_func  strptime
check_func  sched_getaffinity
check_func  sysconf
check_func  sysctl
```

check_func()的定义如下所示。

```python
check_func(){
    log check_func "$@"
    func=$1
    shift
    disable $func
    check_ld "cc" "$@" <<EOF && enable $func
extern int $func();
int main(void){ $func(); }
EOF
}
```

从check_func()的定义可以看出，该函数首先将输入的第1个参数赋值给func，然后生成一个下述内容的C语言文件。

```python
extern int $func();
int main(void){ $func(); }
```

最后调用check_ld()完成编译测试。check_ld()的定义如下所示。

```python
check_ld(){
    log check_ld "$@"
    type=$1
    shift 1
    flags=''
    libs=''
    for f; do
        test "${f}" = "${f#-l}" && flags="$flags $f" || libs="$libs $f"
    done
    check_$type $($filter_cflags $flags) || return
    check_cmd $ld $LDFLAGS $flags -o $TMPE $TMPO $libs $extralibs
}
```

其中check_cmd()是个很简单的函数，可以输出日志，如下所示。

```python
check_cmd(){
    log "$@"
    "$@" >> $logfile 2>&1
}
```

例如，“check_func mkstemp”相当于编译了下述代码。

```python
extern int mkstemp();
int main(void){ mkstemp(); }
```

check_header()

check_header()用于检查头文件。Configure中与check_header()有关的代码如下所示。

```python
check_header dlfcn.h
check_header dxva2api.h -D_WIN32_WINNT=0x0600
check_header libcrystalhd/libcrystalhd_if.h
check_header malloc.h
check_header poll.h
check_header sys/mman.h
check_header sys/param.h
check_header sys/resource.h
check_header sys/select.h
check_header termios.h
check_header vdpau/vdpau.h
check_header vdpau/vdpau_x11.h
check_header X11/extensions/XvMClib.h
check_header asm/types.h
```

check_header()的定义如下所示。

```python
check_header(){
    log check_header "$@"
    header=$1
    shift
    disable_safe $header
    check_cpp "$@" <<EOF && enable_safe $header
#include <$header>
int x;
EOF
}
```

从check_header()的定义可以看出，该函数首先将输入的第1个参数赋值给header，然后生成一个下述内容的C语言文件。

```python
#include <$header>
int x;
```

最后调用check_cpp()完成编译测试。check_cpp()的定义如下所示。

```python
check_cpp(){
    log check_cpp "$@"
    cat > $TMPC
    log_file $TMPC
    #-E选项，可以让编译器在预处理后停止，并输出预处理结果。
    check_cmd $cc $CPPFLAGS $CFLAGS "$@" -E -o $TMPO $TMPC
}
```

例如，“check_header malloc.h”相当于处理以下C语言文件。

```python
#include <malloc.h>
int x;
```

check_func_headers()

check_func_headers()用于同时检查头文件和函数。Configure中与check_header()有关的代码如下所示。

```python
check_func_headers conio.h kbhit
check_func_headers windows.h PeekNamedPipe
check_func_headers io.h setmode
check_func_headers lzo/lzo1x.h lzo1x_999_compress
check_func_headers windows.h GetProcessAffinityMask
check_func_headers windows.h GetProcessTimes
check_func_headers windows.h MapViewOfFile
check_func_headers windows.h VirtualAlloc
```

check_func_headers()的定义如下所示。

```python
check_func_headers(){
    log check_func_headers "$@"
    headers=$1
    funcs=$2
    shift 2
    {
        for hdr in $headers; do
            echo "#include <$hdr>"
        done
        for func in $funcs; do
            echo "long check_$func(void) { return (long) $func; }"
        done
        echo "int main(void) { return 0; }"
    } | check_ld "cc" "$@" && enable $funcs && enable_safe $headers
}
```

从check_func_headers()的定义可以看出，该函数首先将输入的第1个参数赋值给header，第2个参数赋值给funcs，然后生成一个下述内容的C语言文件。

```python
#include <$hdr>
long check_$func (void) { return (long) $func; }
int main(void) { return 0; }
```

例如，“check_func_headers windows.h PeekNamedPipe”相当于处理以下C语言文件。

```python
#include <windows.h>
long check_PeekNamedPipe (void) { return (long) PeekNamedPipe; }
int main(void) { return 0; }
```

check_mathfunc()

check_mathfunc()用于检查数学类函数。Configure中与check_mathfunc()有关的代码如下所示。

```python
check_mathfunc cbrtf
check_mathfunc exp2
check_mathfunc exp2f
check_mathfunc llrint
check_mathfunc llrintf
check_mathfunc log2
check_mathfunc log2f
check_mathfunc lrint
check_mathfunc lrintf
check_mathfunc round
check_mathfunc roundf
check_mathfunc trunc
check_mathfunc truncf
```

check_mathfunc()的定义如下所示。

```python
check_mathfunc(){
    log check_mathfunc "$@"
    #数学函数名称
    func=$1
    shift
    disable $func
    check_ld "cc" "$@" <<EOF && enable $func
#include <math.h>
float foo(float f) { return $func(f); }
int main(void){ return (int) foo; }
EOF
}
```

从check_mathfunc()的定义可以看出，该函数首先将输入的第1个参数赋值给func，然后生成一个下述内容的C语言文件。

```python
#include <math.h>
float foo(float f) { return $func(f); }
int main(void){ return (int) foo; }
```

最后调用check_ld()完成编译测试。
例如，“check_mathfunc exp2”相当于编译连接了下面这个C文件。

```python
#include <math.h>
float foo(float f) { return exp2(f); }
int main(void){ return (int) foo; }
```

require()

require()用于检查第三方类库。Configure中与require()有关的代码如下所示。

```python
#检查第三方类库
# these are off by default, so fail if requested and not available
#require()函数参数的规范：（名称，头文件，函数名，附加选项）
#require2()函数参数规范类似
enabled avisynth   && require2 vfw32 "windows.h vfw.h" AVIFileInit -lavifil32
enabled frei0r     && { check_header frei0r.h || die "ERROR: frei0r.h header not found"; }
enabled gnutls     && require_pkg_config gnutls gnutls/gnutls.h gnutls_global_init
enabled libaacplus && require  "libaacplus >= 2.0.0" aacplus.h aacplusEncOpen -laacplus
enabled libass     && require_pkg_config libass ass/ass.h ass_library_init
enabled libcelt    && require libcelt celt/celt.h celt_decode -lcelt0 &&
                      { check_lib celt/celt.h celt_decoder_create_custom -lcelt0 ||
                        die "ERROR: libcelt version must be >= 0.11.0."; }
enabled libdc1394  && require_pkg_config libdc1394-2 dc1394/dc1394.h dc1394_new
enabled libdirac   && require_pkg_config dirac                          \
    "libdirac_decoder/dirac_parser.h libdirac_encoder/dirac_encoder.h"  \
    "dirac_decoder_init dirac_encoder_init"
#测试libfaac
enabled libfaac    && require2 libfaac "stdint.h faac.h" faacEncGetVersion -lfaac
enabled libfreetype && require_pkg_config freetype2 "ft2build.h freetype/freetype.h" FT_Init_FreeType
enabled libgsm     && require  libgsm gsm/gsm.h gsm_create -lgsm
enabled libmodplug && require  libmodplug libmodplug/modplug.h ModPlug_Load -lmodplug
enabled libmp3lame && require  "libmp3lame >= 3.98.3" lame/lame.h lame_set_VBR_quality -lmp3lame
enabled libnut     && require  libnut libnut.h nut_demuxer_init -lnut
enabled libopencore_amrnb  && require libopencore_amrnb opencore-amrnb/interf_dec.h Decoder_Interface_init -lopencore-amrnb
enabled libopencore_amrwb  && require libopencore_amrwb opencore-amrwb/dec_if.h D_IF_init -lopencore-amrwb
enabled libopencv  && require_pkg_config opencv opencv/cxcore.h cvCreateImageHeader
enabled libopenjpeg && require libopenjpeg openjpeg.h opj_version -lopenjpeg
enabled libpulse && require_pkg_config libpulse-simple pulse/simple.h pa_simple_new
enabled librtmp    && require_pkg_config librtmp librtmp/rtmp.h RTMP_Socket
enabled libschroedinger && require_pkg_config schroedinger-1.0 schroedinger/schro.h schro_init
enabled libspeex   && require  libspeex speex/speex.h speex_decoder_init -lspeex
enabled libstagefright_h264  && require_cpp libstagefright_h264 "binder/ProcessState.h media/stagefright/MetaData.h
    media/stagefright/MediaBufferGroup.h media/stagefright/MediaDebug.h media/stagefright/MediaDefs.h
    media/stagefright/OMXClient.h media/stagefright/OMXCodec.h" android::OMXClient -lstagefright -lmedia -lutils -lbinder
enabled libtheora  && require  libtheora theora/theoraenc.h th_info_init -ltheoraenc -ltheoradec -logg
enabled libutvideo    && require_cpp utvideo "stdint.h stdlib.h utvideo/utvideo.h utvideo/Codec.h" 'CCodec*' -lutvideo -lstdc++
enabled libv4l2    && require_pkg_config libv4l2 libv4l2.h v4l2_ioctl
enabled libvo_aacenc && require libvo_aacenc vo-aacenc/voAAC.h voGetAACEncAPI -lvo-aacenc
enabled libvo_amrwbenc && require libvo_amrwbenc vo-amrwbenc/enc_if.h E_IF_init -lvo-amrwbenc
enabled libvorbis  && require  libvorbis vorbis/vorbisenc.h vorbis_info_init -lvorbisenc -lvorbis -logg
enabled libvpx     && {
    enabled libvpx_decoder && { check_lib2 "vpx/vpx_decoder.h vpx/vp8dx.h" vpx_codec_dec_init_ver -lvpx ||
                                die "ERROR: libvpx decoder version must be >=0.9.1"; }
    enabled libvpx_encoder && { check_lib2 "vpx/vpx_encoder.h vpx/vp8cx.h" "vpx_codec_enc_init_ver VPX_CQ" -lvpx ||
                                die "ERROR: libvpx encoder version must be >=0.9.6"; } }
#测试libx264
enabled libx264    && require  libx264 x264.h x264_encoder_encode -lx264 &&
                      { check_cpp_condition x264.h "X264_BUILD >= 118" ||
                        die "ERROR: libx264 version must be >= 0.118."; }
enabled libxavs    && require  libxavs xavs.h xavs_encoder_encode -lxavs
enabled libxvid    && require  libxvid xvid.h xvid_global -lxvidcore
enabled openal     && { { for al_libs in "${OPENAL_LIBS}" "-lopenal" "-lOpenAL32"; do
                          check_lib 'AL/al.h' alGetError "${al_libs}" && break; done } ||
                          die "ERROR: openal not found"; } &&
                      { check_cpp_condition "AL/al.h" "defined(AL_VERSION_1_1)" ||
                          die "ERROR: openal version must be 1.1 or compatible"; }
enabled mlib       && require  mediaLib mlib_types.h mlib_VectorSub_S16_U8_Mod -lmlib
enabled openssl    && { check_lib openssl/ssl.h SSL_library_init -lssl -lcrypto ||
                          check_lib openssl/ssl.h SSL_library_init -lssl32 -leay32 ||
                          check_lib openssl/ssl.h SSL_library_init -lssl -lcrypto -lws2_32 -lgdi32 ||
                          die "ERROR: openssl not found"; }
```

由于上述代码量比较大。在这里我们只选择一个典型的例子——libx264来看一下。require()检测libx264的定义如下所示。

```python
#测试libx264
require libx264 x264.h x264_encoder_encode -lx264 &&
  { check_cpp_condition x264.h "X264_BUILD >= 118" ||
    die "ERROR: libx264 version must be >= 0.118."; }
```

从测试libx264的代码可以看出，require()函数的使用方式为：

```plain
require {名称} {头文件} {函数名} {附加选项}
```

require()函数定义如下所示。

```python
#检查依赖项的时候使用
require(){
    name="$1"
    header="$2"
    func="$3"
    shift 3
    check_lib $header $func "$@" || die "ERROR: $name not found"
}
```

从require()的定义可以看出，该函数将第1个参数赋值给name，第2个参数赋值给header，第3个参数赋值给func。最后调用check_lib()函数。
check_lib()的定义如下所示。

```python
#检查类库
check_lib(){
    log check_lib "$@"
    header="$1"
    func="$2"
    shift 2
    check_header $header && check_func $func "$@" && add_extralibs "$@"
}
```

可以看出check_lib()调用了check_header()、check_func()等几个函数完成检查工作。这两个函数在前文中已经介绍过，就不再重复了。
例如检测libx264的时候调用check_header()会生成以下临时文件：

```python
#include <x264.h>
int x;
```

check_cflags ()

check_cflags()用于检查编译器的cflags标志参数。Configure中与check_cflags()有关的代码如下所示。

```python
#添加一些编译选项
# add some useful compiler flags if supported
check_cflags -Wdeclaration-after-statement
check_cflags -Wall
check_cflags -Wno-parentheses
check_cflags -Wno-switch
check_cflags -Wno-format-zero-length
check_cflags -Wdisabled-optimization
check_cflags -Wpointer-arith
check_cflags -Wredundant-decls
check_cflags -Wno-pointer-sign
check_cflags -Wcast-qual
check_cflags -Wwrite-strings
check_cflags -Wtype-limits
check_cflags -Wundef
check_cflags -Wmissing-prototypes
check_cflags -Wno-pointer-to-int-cast
check_cflags -Wstrict-prototypes
check_cflags()函数的定义如下所示。
check_cflags(){
    log check_cflags "$@"
    set -- $($filter_cflags "$@")
    check_cc "$@" <<EOF && append CFLAGS "$@"
int x;
EOF
}
```

从定义可以看出，check_cflags()调用了check_cc()执行命令。整个代码只有一行：

```cpp
int x;
```

check_cc()的定义如下所示。

```python
check_cc(){
    log check_cc "$@"
    cat > $TMPC
    log_file $TMPC
    #很多检查都调用了这个check_cmd
    #-c 只编译不链接
    check_cmd $cc $CPPFLAGS $CFLAGS "$@" -c -o $TMPO $TMPC
}
```

可以看出check_cflags()将输入参数设置到命令行中，并最终调用了check_cmd。
除了上述几个函数之外，还有其他的一些检查编译环境的函数，在这里就不一一列举了。

Echo info

Echo info用于在控制台上打印配置信息。Configure中该部分的代码如下所示。

```python
#在控制台输出信息
echo "install prefix            $prefix"
echo "source path               $source_path"
echo "C compiler                $cc"
echo "ARCH                      $arch ($cpu)"
if test "$build_suffix" != ""; then
    echo "build suffix              $build_suffix"
fi
if test "$progs_suffix" != ""; then
    echo "progs suffix              $progs_suffix"
fi
if test "$extra_version" != ""; then
    echo "version string suffix     $extra_version"
fi
#${}的特异功能：
#${file-my.file.txt}假如 $file 为空值，则使用 my.file.txt 作默认值。(保留没设定及非空值)
#在这里，如果某个变量为空值，则取默认值为no
echo "big-endian                ${bigendian-no}"
echo "runtime cpu detection     ${runtime_cpudetect-no}"
if enabled x86; then
    echo "${yasmexe}                      ${yasm-no}"
    echo "MMX enabled               ${mmx-no}"
    echo "MMX2 enabled              ${mmx2-no}"
    echo "3DNow! enabled            ${amd3dnow-no}"
    echo "3DNow! extended enabled   ${amd3dnowext-no}"
    echo "SSE enabled               ${sse-no}"
    echo "SSSE3 enabled             ${ssse3-no}"
    echo "AVX enabled               ${avx-no}"
    echo "CMOV enabled              ${cmov-no}"
    echo "CMOV is fast              ${fast_cmov-no}"
    echo "EBX available             ${ebx_available-no}"
    echo "EBP available             ${ebp_available-no}"
fi
if enabled arm; then
    echo "ARMv5TE enabled           ${armv5te-no}"
    echo "ARMv6 enabled             ${armv6-no}"
    echo "ARMv6T2 enabled           ${armv6t2-no}"
    echo "ARM VFP enabled           ${armvfp-no}"
    echo "IWMMXT enabled            ${iwmmxt-no}"
    echo "NEON enabled              ${neon-no}"
fi
if enabled mips; then
    echo "MMI enabled               ${mmi-no}"
fi
if enabled ppc; then
    echo "AltiVec enabled           ${altivec-no}"
    echo "PPC 4xx optimizations     ${ppc4xx-no}"
    echo "dcbzl available           ${dcbzl-no}"
fi
if enabled sparc; then
    echo "VIS enabled               ${vis-no}"
fi
echo "debug symbols             ${debug-no}"
echo "strip symbols             ${stripping-no}"
echo "optimize for size         ${small-no}"
echo "optimizations             ${optimizations-no}"
echo "static                    ${static-no}"
echo "shared                    ${shared-no}"
echo "postprocessing support    ${postproc-no}"
echo "new filter support        ${avfilter-no}"
echo "network support           ${network-no}"
echo "threading support         ${thread_type-no}"
echo "safe bitstream reader     ${safe_bitstream_reader-no}"
echo "SDL support               ${sdl-no}"
echo "Sun medialib support      ${mlib-no}"
echo "libdxva2 enabled          ${dxva2-no}"
echo "libva enabled             ${vaapi-no}"
```

```
echo "libvdpau enabled          ${vdpau-no}"
echo "AVISynth enabled          ${avisynth-no}"
echo "frei0r enabled            ${frei0r-no}"
echo "gnutls enabled            ${gnutls-no}"
echo "libaacplus enabled        ${libaacplus-no}"
echo "libass enabled            ${libass-no}"
echo "libcdio support           ${libcdio-no}"
echo "libcelt enabled           ${libcelt-no}"
echo "libdc1394 support         ${libdc1394-no}"
echo "libdirac enabled          ${libdirac-no}"
echo "libfaac enabled           ${libfaac-no}"
echo "libgsm enabled            ${libgsm-no}"
echo "libmodplug enabled        ${libmodplug-no}"
echo "libmp3lame enabled        ${libmp3lame-no}"
echo "libnut enabled            ${libnut-no}"
echo "libopencore-amrnb support ${libopencore_amrnb-no}"
echo "libopencore-amrwb support ${libopencore_amrwb-no}"
echo "libopencv support         ${libopencv-no}"
echo "libopenjpeg enabled       ${libopenjpeg-no}"
echo "libpulse enabled          ${libpulse-no}"
echo "librtmp enabled           ${librtmp-no}"
echo "libschroedinger enabled   ${libschroedinger-no}"
echo "libspeex enabled          ${libspeex-no}"
echo "libstagefright-h264 enabled    ${libstagefright_h264-no}"
echo "libtheora enabled         ${libtheora-no}"
echo "libutvideo enabled        ${libutvideo-no}"
echo "libv4l2 enabled           ${libv4l2-no}"
echo "libvo-aacenc support      ${libvo_aacenc-no}"
echo "libvo-amrwbenc support    ${libvo_amrwbenc-no}"
echo "libvorbis enabled         ${libvorbis-no}"
echo "libvpx enabled            ${libvpx-no}"
echo "libx264 enabled           ${libx264-no}"
echo "libxavs enabled           ${libxavs-no}"
echo "libxvid enabled           ${libxvid-no}"
echo "openal enabled            ${openal-no}"
echo "openssl enabled           ${openssl-no}"
echo "zlib enabled              ${zlib-no}"
echo "bzlib enabled             ${bzlib-no}"
echo

for type in decoder encoder hwaccel parser demuxer muxer protocol filter bsf indev outdev; do
    echo "Enabled ${type}s:"
    eval list=\$$(toupper $type)_LIST
    print_enabled '_*' $list | sort | pr -r -3 -t
    echo
done

license="LGPL version 2.1 or later"
if enabled nonfree; then
    license="nonfree and unredistributable"
elif enabled gplv3; then
    license="GPL version 3 or later"
elif enabled lgplv3; then
    license="LGPL version 3 or later"
elif enabled gpl; then
    license="GPL version 2 or later"
fi

echo "License: $license"
```

有关这段代码，有一个地方需要注意：很多的${}符号中的字符为“XXX-no”，这种格式的意思是如果XXX取值为空，则使用默认值“no”（这个规则比较奇特）。

Write basic info

Write basic info用于向config.mak中写入一些基本信息。Configure中该部分的代码如下所示。

```python
#创建config.mak和config.h
#根据情况也会创建config.asm
echo "Creating config.mak and config.h..."

test -e Makefile || $ln_s "$source_path/Makefile" .

enabled stripping || strip="echo skipping strip"
#重要：需要输出的文件
#TMPH是一个临时文件，最终会拷贝给config.h
config_files="$TMPH config.mak"
#写入config.mak文件
#首先写入一些基本信息
#"<<EOF"表示后续的输入作为子命令或子shell的输入，直到遇到"EOF"，再次返回到
#主调shell，可将其理解为分界符（delimiter）。
#最后的"EOF"必须单独占一行
cat > config.mak <<EOF
# Automatically generated by configure - do not modify!
ifndef FFMPEG_CONFIG_MAK
FFMPEG_CONFIG_MAK=1
FFMPEG_CONFIGURATION=$FFMPEG_CONFIGURATION
prefix=$prefix
```

```
prefix=$prefix
LIBDIR=\$(DESTDIR)$libdir
SHLIBDIR=\$(DESTDIR)$shlibdir
INCDIR=\$(DESTDIR)$incdir
BINDIR=\$(DESTDIR)$bindir
DATADIR=\$(DESTDIR)$datadir
MANDIR=\$(DESTDIR)$mandir
SRC_PATH=$source_path
ifndef MAIN_MAKEFILE
SRC_PATH:=\$(SRC_PATH:.%=..%)
endif
CC_IDENT=$cc_ident
ARCH=$arch
CC=$cc
CXX=$cxx
AS=$as
LD=$ld
DEPCC=$dep_cc
YASM=$yasmexe
YASMDEP=$yasmexe
AR=$ar
RANLIB=$ranlib
CP=cp -p
LN_S=$ln_s
STRIP=$strip
CPPFLAGS=$CPPFLAGS
CFLAGS=$CFLAGS
CXXFLAGS=$CXXFLAGS
ASFLAGS=$ASFLAGS
AS_O=$CC_O
CC_O=$CC_O
CXX_O=$CXX_O
LDFLAGS=$LDFLAGS
FFSERVERLDFLAGS=$FFSERVERLDFLAGS
SHFLAGS=$SHFLAGS
YASMFLAGS=$YASMFLAGS
BUILDSUF=$build_suffix
PROGSSUF=$progs_suffix
FULLNAME=$FULLNAME
LIBPREF=$LIBPREF
LIBSUF=$LIBSUF
LIBNAME=$LIBNAME
SLIBPREF=$SLIBPREF
SLIBSUF=$SLIBSUF
EXESUF=$EXESUF
EXTRA_VERSION=$extra_version
DEPFLAGS=$DEPFLAGS
CCDEP=$CCDEP
CXXDEP=$CXXDEP
ASDEP=$ASDEP
CC_DEPFLAGS=$CC_DEPFLAGS
AS_DEPFLAGS=$AS_DEPFLAGS
HOSTCC=$host_cc
HOSTCFLAGS=$host_cflags
HOSTEXESUF=$HOSTEXESUF
HOSTLDFLAGS=$host_ldflags
HOSTLIBS=$host_libs
TARGET_EXEC=$target_exec
TARGET_PATH=$target_path
SDL_LIBS=$sdl_libs
SDL_CFLAGS=$sdl_cflags
LIB_INSTALL_EXTRA_CMD=$LIB_INSTALL_EXTRA_CMD
EXTRALIBS=$extralibs
INSTALL=$install
LIBTARGET=${LIBTARGET}
SLIBNAME=${SLIBNAME}
SLIBNAME_WITH_VERSION=${SLIBNAME_WITH_VERSION}
SLIBNAME_WITH_MAJOR=${SLIBNAME_WITH_MAJOR}
SLIB_CREATE_DEF_CMD=${SLIB_CREATE_DEF_CMD}
SLIB_EXTRA_CMD=${SLIB_EXTRA_CMD}
SLIB_INSTALL_NAME=${SLIB_INSTALL_NAME}
SLIB_INSTALL_LINKS=${SLIB_INSTALL_LINKS}
SLIB_INSTALL_EXTRA_LIB=${SLIB_INSTALL_EXTRA_LIB}
SLIB_INSTALL_EXTRA_SHLIB=${SLIB_INSTALL_EXTRA_SHLIB}
SAMPLES:=${samples:-\$(FATE_SAMPLES)}
NOREDZONE_FLAGS=$noredzone_flags
EOF
#获取版本
#主要通过各个类库文件夹中的version.h文件
#读取XXX_VERSION（相当于把头文件当成一个文本来读）
get_version(){
    name=$1
    file=$source_path/$2
# This condition will be removed when we stop supporting old libpostproc versions
if ! test "$name" = LIBPOSTPROC || test "$postproc_version" = current; then
    eval $(grep "#define ${name}_VERSION_M" "$file" | awk '{ print $2"="$3 }')
    eval ${name}_VERSION=\${name}_VERSION_MAJOR.\${name}_VERSION_MINOR.\${name}_VERSION_MICRO
fi
    lcname=$(tolower $name)
    eval echo "${lcname}_VERSION=\${name}_VERSION" >> config.mak
    eval echo "${lcname}_VERSION_MAJOR=\${name}_VERSION_MAJOR" >> config.mak
}
```

```
#获取版本
get_version LIBAVCODEC  libavcodec/version.h
get_version LIBAVDEVICE libavdevice/avdevice.h
get_version LIBAVFILTER libavfilter/version.h
get_version LIBAVFORMAT libavformat/version.h
get_version LIBAVUTIL   libavutil/avutil.h
get_version LIBPOSTPROC libpostproc/postprocess.h
get_version LIBSWRESAMPLE libswresample/swresample.h
get_version LIBSWSCALE  libswscale/swscale.h
```

关于这段代码，有以下几点需要注意：
（1）“cat > config.mak <<EOF”的作用就是往config.mak中写入文本，当遇到“EOF”的时候写入结束
（2）get_version()用于获取当前的FFmpeg源代码中各个类库的版本。通过把各个类库文件夹下的version.h当作文本读取之后，分析字符串并且得到版本号，最终写入config.mak文件。

print_config()

print_config()用于向config.h、config.mak、config.asm中写入所有配置信息。Configure中该部分的代码如下所示。

```python
#输出所有的配置信息包含3类：
#以“ARCH_”开头，包含系统架构信息
#以“HAVE_”开头，包含系统特征信息
#以“CONFIG_”开头，包含编译配置（数量最多，包含协议、复用器、编解码器等的配置，将近1000行）
#config_files
print_config ARCH_   "$config_files" $ARCH_LIST
print_config HAVE_   "$config_files" $HAVE_LIST
print_config CONFIG_ "$config_files" $CONFIG_LIST       \
                                     $CONFIG_EXTRA      \
                                     $ALL_COMPONENTS    \
```

从源代码中可以看出，其中调用了一个函数print_config()。print_config()的源代码如下所示。

```python
# 输出文本到config.mak，config.h等文件
# 该函数的示例调用方法：print_config CONFIG_ "$config_files" $CONFIG_LIST
print_config(){
# 前缀
    pfx=$1
# 文件列表
    files=$2
    # 位置参数可以用shift命令左移。比如shift 3表示原来的$4现在变成$1
    shift 2
    #for循环中，当没有in指定列表时，for会默认取命令行参数列表。
    #在这里取的就是$CONFIG_LIST 等
    for cfg; do
    # toupper()：转换为大写
        ucname="$(toupper $cfg)"
        # files= config.h config.mak config.asm
        # 循环输出
        for f in $files; do
        # "x#*/"代表去取x的第一个slash之后的所有内容（不包括slash）
        # "#"代表删除从前往后最小匹配的内容
        # "f##*."代表去取f的第一个"."之后的所有内容。在这里是“h”、“mak”等
        # 在这里print_config_h(),print_config_mak(),print_config_asm()
            "print_config_${f##*.}" $cfg ${pfx}${ucname} >>$f
        done
    done
}
```

可以看出print_config()的第1个参数是写入参数的前缀（例如可以取“ARCH_”、“HAVE_”、“CONFIG_”）；第2个参数是文件列表（例如可以是“config.h config.mak config.asm”）；第3个以后的的参数就是需要写入的变量（例如$ARCH_LIST、$CONFIG_LIST等）。
print_config()有两层循环：外层循环遍历了所有的变量（例如$CONFIG_LIST），内层循环遍历了所有文件（例如“config.h config.mak”），其中调用了一个函数print_config_XXX()，其中“XXX”根据文件后缀的不同可以取不同的值（例如“h”、“mak”）。下面举例看两个函数：print_config_h()和print_config_mak()。

print_config_h()

print_config_h()用于输出配置信息至config.h。该函数的源代码如下所示。

```python
#输出config.h的时候使用
#调用示例：print_config_h ffplay CONFIG_FFPLAY
print_config_h(){
#command1  && command2
#&&左边的命令（命令1）返回真(即返回0，成功被执行）后，&&右边的命令（命令2）才能够被执行
#command1 || command2
#||左边的命令（命令1）未执行成功，那么就执行||右边的命令（命令2）
    enabled $1 && v=1 || v=0
#示例：#define CONFIG_FFPLAY 1
    echo "#define $2 $v"
}
```

从源代码中可以看出，参数1是变量名称，参数2是经过处理后准备写入文件的变量名称（变量名转换成了大写并且添加了前缀）。如果参数1所指向的变量是enabled的，那么v取值为1，那么写入文件的格式就是：

```cpp
#define {处理后变量名称} 1
```

如果参数1所指向的变量不是enabled的，那么v取值为0，那么写入文件的格式就是：

```python
#define {处理后变量名称} 0
```

print_config_mak()

print_config_mak()用于输出配置信息至config.mak。该函数的源代码如下所示。

```python
#输出config.mak的时候使用
print_config_mak(){
    enabled $1 && v= || v=!
    echo "$v$2=yes"
}
```

从源代码中可以看出print_config_mak()的原理和print_config_h()是类似的。如果变量是enabled的，那么写入文件的格式就是：

```plain
{处理后变量名称}=yes
```

如果变量不是enabled的，那么写入文件的格式就是：

```plain
!{处理后变量名称}=yes
```

print_enabled()

print_enabled()用于向config.mak写入所有enabled的组件信息。这方面功能通过print_enabled()函数完成，就不再详细分析了。

pkgconfig_generate()

pkgconfig_generate()用于向libavXXX/libavXXX.pc中写入pkgconfig信息（XXX代表avcodec，avformat等）。这方面的代码还没有细看，以后有机会再进行补充。

源代码（包含注释）

至此，FFmpeg的Configure的流程就大致梳理完毕了，最后附上和Configure有关的config.mak、config.h以及Configure本身的源代码。

config.mak源代码

```python
# FFmpeg config.mak
#
# 注释：雷霄骅
# leixiaohua1020@126.com
# http://blog.csdn.net/leixiaohua1020
#
# Configure脚本生成的Makefile，包含了各种配置信息。
#
# Automatically generated by configure - do not modify!
#基本信息
ifndef FFMPEG_CONFIG_MAK
```

```
FFMPEG_CONFIG_MAK=1
FFMPEG_CONFIGURATION=
#各种路径=======================================
prefix=/usr/local
LIBDIR=$(DESTDIR)${prefix}/lib
SHLIBDIR=$(DESTDIR)${prefix}/bin
INCDIR=$(DESTDIR)${prefix}/include
BINDIR=$(DESTDIR)${prefix}/bin
DATADIR=$(DESTDIR)${prefix}/share/ffmpeg
MANDIR=$(DESTDIR)${prefix}/share/man
#是个相对路径
SRC_PATH=.
ifndef MAIN_MAKEFILE
SRC_PATH:=$(SRC_PATH:.%=..%)
endif
#工具集=======================================
CC_IDENT=gcc 4.6.2 (GCC)
#架构
ARCH=x86
#编译器
CC=gcc
CXX=g++
AS=gcc
#链接器
LD=gcc
DEPCC=gcc
#汇编器
YASM=yasm
YASMDEP=yasm
#生成静态库.a工具
AR=ar
RANLIB=ranlib
CP=cp -p
LN_S=ln -sf
STRIP=strip
#参数集=======================================
#编译器的参数
CPPFLAGS= -D_ISOC99_SOURCE -D_FILE_OFFSET_BITS=64 -D_LARGEFILE_SOURCE -U__STRICT_ANSI__
CFLAGS=   -std=c99 -fno-common -fomit-frame-pointer -I/include/SDL -D_GNU_SOURCE=1 -Dmain=SDL_main -g -Wdeclaration-after-statement -Wall -Wno-parentheses -Wno-switch -Wno-format-zero-length -Wdisabled-optimization -Wpointer-arith -Wredundant-decls -Wno-pointer-sign -Wcast-qual -Wwrite-strings -Wtype-limits -Wundef -Wmissing-prototypes -Wno-pointer-to-int-cast -Wstrict-prototypes -O3 -fno-math-errno -fno-signed-zeros -fno-tree-vectorize -Werror=implicit-function-declaration -Werror=missing-prototypes
CXXFLAGS=  -D__STDC_CONSTANT_MACROS
ASFLAGS=   -g
#目标文件有关的参数
AS_O=-o $@
CC_O=-o $@
CXX_O=-o $@
#链接器有关的参数
LDFLAGS= -Wl,--as-needed -Wl,--warn-common -Wl,-rpath-link=libpostproc:libswresample:libswscale:libavfilter:libavdevice:libavformat:libavcodec:libavutil
FFSERVERLDFLAGS=-Wl,-E
SHFLAGS=-shared -Wl,--output-def,$$(@:$(SLIBSUF)=.def) -Wl,--out-implib,$(SUBDIR)lib$(SLIBNAME:$(SLIBSUF)=.dll.a) -Wl,--enable-runtime-pseudo-reloc -Wl,--enable-auto-image-base -Wl,-Bsymbolic -Wl,--version-script,$(SUBDIR)lib$(NAME).ver
YASMFLAGS=-f win32  -DPREFIX
#前缀后缀=======================================
BUILDSUF=
PROGSSUF=
#${NAME}位于每个liavXXX/Makefile中，例如avformat
FULLNAME=$(NAME)$(BUILDSUF)
LIBPREF=lib
LIBSUF=.a
#例如libavformat.a
LIBNAME=$(LIBPREF)$(FULLNAME)$(LIBSUF)
SLIBPREF=
SLIBSUF=.dll
EXESUF=.exe
EXTRA_VERSION=
DEPFLAGS=$(CPPFLAGS) $(CFLAGS) -MM
CCDEP=
CXXDEP=$(DEPCC) $(DEPFLAGS) $< | sed -e "/^\#.*/d" -e "s,^[[:space:]]*$(*F)\\.o,$(@D)/$(*F).o," > $(@:.o=.d)
ASDEP=
CC_DEPFLAGS=-MMD -MF $(@:.o=.d) -MT $@
AS_DEPFLAGS=-MMD -MF $(@:.o=.d) -MT $@
HOSTCC=gcc
HOSTCFLAGS=-D_ISOC99_SOURCE -O3 -g -std=c99 -Wall
HOSTEXESUF=.exe
HOSTLDFLAGS=
HOSTLIBS=-lm
TARGET_EXEC=
TARGET_PATH=$(CURDIR)
#SDL
SDL_LIBS=-L/lib -lmingw32 -lSDLmain -lSDL -mwindows
SDL_CFLAGS=-I/include/SDL -D_GNU_SOURCE=1 -Dmain=SDL_main
LIB_INSTALL_EXTRA_CMD=$$(RANLIB) "$(LIBDIR)/$(LIBNAME)"
#链接
EXTRALIBS=-lavicap32 -lws2_32 -L/lib -lmingw32 -lSDLmain -lSDL -mwindows -lm -lz -lpsapi
INSTALL=install
LIBTARGET=i386
#例如libavformat.dll
SLIBNAME=$(SLIBPREF)$(FULLNAME)$(SLIBSUF)
```

```
#LIBVERSION变量位于library.mak
#例如libavformat-53.dll
#生成的Dll似乎就是这个版本的
SLIBNAME_WITH_VERSION=$(SLIBPREF)$(FULLNAME)-$(LIBVERSION)$(SLIBSUF)
#例如libavformat-53.31.100.dll
SLIBNAME_WITH_MAJOR=$(SLIBPREF)$(FULLNAME)-$(LIBMAJOR)$(SLIBSUF)
SLIB_CREATE_DEF_CMD=
#生成导出库lib，会调用lib.exe
SLIB_EXTRA_CMD=-lib.exe /machine:$(LIBTARGET) /def:$$(@:$(SLIBSUF)=.def) /out:$(SUBDIR)$(SLIBNAME:$(SLIBSUF)=.lib)
SLIB_INSTALL_NAME=$(SLIBNAME_WITH_MAJOR)
SLIB_INSTALL_LINKS=
SLIB_INSTALL_EXTRA_LIB=lib$(SLIBNAME:$(SLIBSUF)=.dll.a) $(SLIBNAME_WITH_MAJOR:$(SLIBSUF)=.def)
SLIB_INSTALL_EXTRA_SHLIB=$(SLIBNAME:$(SLIBSUF)=.lib)
SAMPLES:=$(FATE_SAMPLES)
NOREDZONE_FLAGS=-mno-red-zone
#版本信息=======================================
libavcodec_VERSION=53.60.100
libavcodec_VERSION_MAJOR=53
libavdevice_VERSION=53.4.100
libavdevice_VERSION_MAJOR=53
libavfilter_VERSION=2.60.100
libavfilter_VERSION_MAJOR=2
libavformat_VERSION=53.31.100
libavformat_VERSION_MAJOR=53
libavutil_VERSION=51.34.101
libavutil_VERSION_MAJOR=51
libpostproc_VERSION=52.0.100
libpostproc_VERSION_MAJOR=52
libswresample_VERSION=0.6.100
libswresample_VERSION_MAJOR=0
libswscale_VERSION=2.1.100
libswscale_VERSION_MAJOR=2
#组件配置=======================================
#ARCH_
!ARCH_ALPHA=yes
!ARCH_ARM=yes
!ARCH_AVR32=yes
!ARCH_AVR32_AP=yes
!ARCH_AVR32_UC=yes
!ARCH_BFIN=yes
!ARCH_IA64=yes
!ARCH_M68K=yes
!ARCH_MIPS=yes
!ARCH_MIPS64=yes
!ARCH_PARISC=yes
!ARCH_PPC=yes
!ARCH_PPC64=yes
!ARCH_S390=yes
!ARCH_SH4=yes
!ARCH_SPARC=yes
!ARCH_SPARC64=yes
!ARCH_TOMI=yes
ARCH_X86=yes
ARCH_X86_32=yes
!ARCH_X86_64=yes
#HAVE_
!HAVE_ALTIVEC=yes
HAVE_AMD3DNOW=yes
HAVE_AMD3DNOWEXT=yes
!HAVE_ARMV5TE=yes
!HAVE_ARMV6=yes
!HAVE_ARMV6T2=yes
!HAVE_ARMVFP=yes
HAVE_AVX=yes
!HAVE_IWMMXT=yes
!HAVE_MMI=yes
HAVE_MMX=yes
HAVE_MMX2=yes
//略……
HAVE_YASM=yes
#CONFIG_
CONFIG_BSFS=yes
CONFIG_DECODERS=yes
CONFIG_DEMUXERS=yes
CONFIG_ENCODERS=yes
CONFIG_FILTERS=yes
!CONFIG_HWACCELS=yes
CONFIG_INDEVS=yes
CONFIG_MUXERS=yes
CONFIG_OUTDEVS=yes
CONFIG_PARSERS=yes
CONFIG_PROTOCOLS=yes
CONFIG_FFPLAY=yes
CONFIG_FFPROBE=yes
!CONFIG_FFSERVER=yes
CONFIG_FFMPEG=yes
!CONFIG_AVPLAY=yes
!CONFIG_AVPROBE=yes
!CONFIG_AVSERVER=yes
CONFIG_AANDCT=yes
CONFIG_AC3DSP=yes
```

```
CONFIG_AVCODEC=yes
CONFIG_AVDEVICE=yes
CONFIG_AVFILTER=yes
CONFIG_AVFORMAT=yes
!CONFIG_AVISYNTH=yes
!CONFIG_BZLIB=yes
!CONFIG_CRYSTALHD=yes
CONFIG_DCT=yes
!CONFIG_DOC=yes
CONFIG_DWT=yes
!CONFIG_DXVA2=yes
CONFIG_FASTDIV=yes
CONFIG_FFT=yes
!CONFIG_FREI0R=yes
!CONFIG_GNUTLS=yes
CONFIG_GOLOMB=yes
!CONFIG_GPL=yes
!CONFIG_GRAY=yes
CONFIG_H264CHROMA=yes
CONFIG_H264DSP=yes
CONFIG_H264PRED=yes
!CONFIG_HARDCODED_TABLES=yes
CONFIG_HUFFMAN=yes
!CONFIG_LIBAACPLUS=yes
!CONFIG_LIBASS=yes
!CONFIG_LIBCDIO=yes
!CONFIG_LIBCELT=yes
!CONFIG_LIBDC1394=yes
!CONFIG_LIBDIRAC=yes
!CONFIG_LIBFAAC=yes
!CONFIG_LIBFREETYPE=yes
!CONFIG_LIBGSM=yes
!CONFIG_LIBMODPLUG=yes
!CONFIG_LIBMP3LAME=yes
!CONFIG_LIBNUT=yes
!CONFIG_LIBOPENCORE_AMRNB=yes
!CONFIG_LIBOPENCORE_AMRWB=yes
!CONFIG_LIBOPENCV=yes
!CONFIG_LIBOPENJPEG=yes
!CONFIG_LIBPULSE=yes
!CONFIG_LIBRTMP=yes
!CONFIG_LIBSCHROEDINGER=yes
!CONFIG_LIBSPEEX=yes
!CONFIG_LIBSTAGEFRIGHT_H264=yes
!CONFIG_LIBTHEORA=yes
!CONFIG_LIBUTVIDEO=yes
!CONFIG_LIBV4L2=yes
!CONFIG_LIBVO_AACENC=yes
!CONFIG_LIBVO_AMRWBENC=yes
!CONFIG_LIBVORBIS=yes
!CONFIG_LIBVPX=yes
!CONFIG_LIBX264=yes
!CONFIG_LIBXAVS=yes
!CONFIG_LIBXVID=yes
#此处省略将近1000条…
CONFIG_RTMP_PROTOCOL=yes
CONFIG_RTMPT_PROTOCOL=yes
CONFIG_RTMPE_PROTOCOL=yes
CONFIG_RTMPTE_PROTOCOL=yes
CONFIG_RTMPS_PROTOCOL=yes
CONFIG_RTP_PROTOCOL=yes
CONFIG_TCP_PROTOCOL=yes
!CONFIG_TLS_PROTOCOL=yes
CONFIG_UDP_PROTOCOL=yes
#Test
ACODEC_TESTS=ac3_fixed adpcm_adx adpcm_ima_qt adpcm_ima_wav adpcm_ms adpcm_swf adpcm_yam alac aref flac g722 g723_1 g726 mp2 pcm_alaw m_f32be pcm_f32le pcm_f64be pcm_f64le pcm_mulaw pcm_s16be pcm_s16le pcm_s24be pcm_s24daud pcm_s24le pcm_s32be pcm_s32le pcm_s8 pcm_u8 av1 wmav2
VCODEC_TESTS=amv asv1 asv2 cljr dnxhd_1080i dnxhd_720p dnxhd_720p_10bit dnxhd_720p_rd dv dv50 dv_411 error ffv1 flashsv flashsv2 flv 1 h263 h263p huffyuv jpeg2000 jpegls ljpeg mjpeg mpeg mpeg1b mpeg2 mpeg2_422 mpeg2_idct_int mpeg2_ilace mpeg2_ivlc_qprd mpeg2thread m 2thread_ilace mpeg4 mpeg4_adap mpeg4_qpel mpeg4_qprd mpeg4adv mpeg4nr mpeg4thread mpng msmpeg4 msmpeg4v2 msvideo1 prores qtrle qtrleg rc rgb roq rv10 rv20 snow snowll svq1 v210 vref wmv1 wmv2 yuv zlib zmbv
LAVF_TESTS=aiff alaw asf au avi bmp caf dpx dv_fmt ffm flv_fmt gif gxf jpg mkv mmf mov mpg mulaw mxf mxf_d10 nut ogg pbmpipe pcx pgm pipe pixfmt png ppm ppmpipe rm rso sgi sox swf tga tiff ts voc voc_s16 wav wtv yuv4mpeg
LAVFI_TESTS=crop crop_scale crop_scale_vflip crop_vflip null pixdesc pixfmts_copy pixfmts_crop pixfmts_hflip pixfmts_null pixfmts_pad xfmts_scale pixfmts_vflip scale200 scale500 vflip vflip_crop vflip_vflip
SEEK_TESTS=seek_ac3_rm seek_adpcm_ima_wav seek_adpcm_ms_wav seek_adpcm_qt_aiff seek_adpcm_swf_flv seek_adpcm_yam_wav seek_alac_m4a se asv1_avi seek_asv2_avi seek_dnxhd_1080i_mov seek_dnxhd_720p_dnxhd seek_dnxhd_720p_rd_dnxhd seek_dv411_dv seek_dv50_dv seek_dv_dv seek ror_mpeg4_adv_avi seek_ffv1_avi seek_flac_flac seek_flashsv_flv seek_flv_flv seek_g726_wav seek_h261_avi seek_h263_avi seek_h263p_avi ek_huffyuv_avi seek_image_bmp seek_image_jpg seek_image_pcx seek_image_pgm seek_image_ppm seek_image_sgi seek_image_tga seek_image_ti seek_jpegls_avi seek_lavf_aif seek_lavf_al seek_lavf_asf seek_lavf_au seek_lavf_avi seek_lavf_dv seek_lavf_ffm seek_lavf_flv seek_lav if seek_lavf_gxf seek_lavf_mkv seek_lavf_mmf seek_lavf_mov seek_lavf_mpg seek_lavf_mxf seek_lavf_mxf_d10 seek_lavf_nut seek_lavf_ogg k_lavf_rm seek_lavf_swf seek_lavf_ts seek_lavf_ul seek_lavf_voc seek_lavf_wav seek_lavf_wtv seek_lavf_y4m seek_ljpeg_avi seek_mjpeg_a seek_mp2_mp2 seek_mpeg1_mpg seek_mpeg1b_mpg seek_mpeg2_422_mpg seek_mpeg2_idct_int_mpg seek_mpeg2i_mpg seek_mpeg2ivlc_qprd_mpg seek_m 2reuse_mpg seek_mpeg2thread_mpg seek_mpeg2threadivlc_mpg seek_mpeg4_adap_avi seek_mpeg4_adv_avi seek_mpeg4_nr_avi seek_mpeg4_qprd_avi ek_mpeg4_rc_avi seek_mpeg4_thread_avi seek_msmpeg4_avi seek_msmpeg4v2_avi seek_odivx_mp4 seek_pbmpipe_pbm seek_pcm_alaw_wav seek_pcm_ be_au seek_pcm_f32le_wav seek_pcm_f64be_au seek_pcm_f64le_wav seek_pcm_mulaw_wav seek_pcm_s16be_mov seek_pcm_s16le_wav seek_pcm_s24be v seek_pcm_s24daud_302 seek_pcm_s24le_wav seek_pcm_s32be_mov seek_pcm_s32le_wav seek_pcm_s8_mov seek_pcm_u8_wav seek_pgmpipe_pgm seek mpipe_ppm seek_rgb_avi seek_roqav_roq seek_rv10_rm seek_rv20_rm seek_snow53_avi seek_snow_avi seek_svq1_mov seek_wmav1_asf seek_wmav2 f seek_wmv1_avi seek_wmv2_avi seek_yuv_avi
endif # FFMPEG_CONFIG_MAK
```

config.h源代码

```cpp
/**
 * FFmpeg config.h
 *
 * 注释：雷霄骅
 * leixiaohua1020@126.com
 * http://blog.csdn.net/leixiaohua1020
 *
 * Configure脚本生成的config.h，包含了各种配置信息。
 /
/* Automatically generated by configure - do not modify! */
#ifndef FFMPEG_CONFIG_H
#define FFMPEG_CONFIG_H
#define FFMPEG_CONFIGURATION ""
#define FFMPEG_LICENSE "LGPL version 2.1 or later"
#define FFMPEG_DATADIR "/usr/local/share/ffmpeg"
#define AVCONV_DATADIR "/usr/local/share/ffmpeg"
#define CC_TYPE "gcc"
#define CC_VERSION __VERSION__
#define restrict restrict
#define EXTERN_PREFIX "_"
#define EXTERN_ASM _
#define SLIBSUF ".dll"
#define ARCH_ALPHA 0
#define ARCH_ARM 0
#define ARCH_AVR32 0
#define ARCH_AVR32_AP 0
#define ARCH_AVR32_UC 0
#define ARCH_BFIN 0
#define ARCH_IA64 0
#define ARCH_M68K 0
#define ARCH_MIPS 0
#define ARCH_MIPS64 0
#define ARCH_PARISC 0
#define ARCH_PPC 0
#define ARCH_PPC64 0
#define ARCH_S390 0
#define ARCH_SH4 0
#define ARCH_SPARC 0
#define ARCH_SPARC64 0
#define ARCH_TOMI 0
#define ARCH_X86 1
#define ARCH_X86_32 1
#define ARCH_X86_64 0
#define HAVE_ALTIVEC 0
#define HAVE_AMD3DNOW 1
#define HAVE_AMD3DNOWEXT 1
#define HAVE_ARMV5TE 0
#define HAVE_ARMV6 0
#define HAVE_ARMV6T2 0
#define HAVE_ARMVFP 0
#define HAVE_AVX 1
#define HAVE_IWMMXT 0
#define HAVE_MMI 0
#define HAVE_MMX 1
#define HAVE_MMX2 1
//略……
#define HAVE_YASM 1
#define CONFIG_BSFS 1
#define CONFIG_DECODERS 1
#define CONFIG_DEMUXERS 1
#define CONFIG_ENCODERS 1
#define CONFIG_FILTERS 1
#define CONFIG_HWACCELS 0
#define CONFIG_INDEVS 1
#define CONFIG_MUXERS 1
#define CONFIG_OUTDEVS 1
#define CONFIG_PARSERS 1
#define CONFIG_PROTOCOLS 1
#define CONFIG_FFPLAY 1
#define CONFIG_FFPROBE 1
#define CONFIG_FFSERVER 0
#define CONFIG_FFMPEG 1
#define CONFIG_AVPLAY 0
#define CONFIG_AVPROBE 0
#define CONFIG_AVSERVER 0
#define CONFIG_AANDCT 1
#define CONFIG_AC3DSP 1
#define CONFIG_AVCODEC 1
#define CONFIG_AVDEVICE 1
#define CONFIG_AVFILTER 1
#define CONFIG_AVFORMAT 1
#define CONFIG_AVISYNTH 0
```

```
#define CONFIG_BZLIB 0
#define CONFIG_CRYSTALHD 0
#define CONFIG_DCT 1
#define CONFIG_DOC 0
#define CONFIG_DWT 1
#define CONFIG_DXVA2 0
#define CONFIG_FASTDIV 1
#define CONFIG_FFT 1
#define CONFIG_FREI0R 0
#define CONFIG_GNUTLS 0
#define CONFIG_GOLOMB 1
#define CONFIG_GPL 0
#define CONFIG_GRAY 0
#define CONFIG_H264CHROMA 1
#define CONFIG_H264DSP 1
#define CONFIG_H264PRED 1
#define CONFIG_HARDCODED_TABLES 0
#define CONFIG_HUFFMAN 1
#define CONFIG_LIBAACPLUS 0
#define CONFIG_LIBASS 0
#define CONFIG_LIBCDIO 0
#define CONFIG_LIBCELT 0
#define CONFIG_LIBDC1394 0
#define CONFIG_LIBDIRAC 0
#define CONFIG_LIBFAAC 0
#define CONFIG_LIBFREETYPE 0
#define CONFIG_LIBGSM 0
#define CONFIG_LIBMODPLUG 0
#define CONFIG_LIBMP3LAME 0
#define CONFIG_LIBNUT 0
#define CONFIG_LIBOPENCORE_AMRNB 0
#define CONFIG_LIBOPENCORE_AMRWB 0
#define CONFIG_LIBOPENCV 0
#define CONFIG_LIBOPENJPEG 0
#define CONFIG_LIBPULSE 0
#define CONFIG_LIBRTMP 0
#define CONFIG_LIBSCHROEDINGER 0
#define CONFIG_LIBSPEEX 0
#define CONFIG_LIBSTAGEFRIGHT_H264 0
#define CONFIG_LIBTHEORA 0
#define CONFIG_LIBUTVIDEO 0
#define CONFIG_LIBV4L2 0
#define CONFIG_LIBVO_AACENC 0
#define CONFIG_LIBVO_AMRWBENC 0
#define CONFIG_LIBVORBIS 0
#define CONFIG_LIBVPX 0
#define CONFIG_LIBX264 0
//此处省略将近1000条
#define CONFIG_RTMP_PROTOCOL 1
#define CONFIG_RTMPT_PROTOCOL 1
#define CONFIG_RTMPE_PROTOCOL 1
#define CONFIG_RTMPTE_PROTOCOL 1
#define CONFIG_RTMPS_PROTOCOL 1
#define CONFIG_RTP_PROTOCOL 1
#define CONFIG_TCP_PROTOCOL 1
#define CONFIG_TLS_PROTOCOL 0
#define CONFIG_UDP_PROTOCOL 1
#endif /* FFMPEG_CONFIG_H */
```

Configure的源代码

```
[plain]
#!/bin/sh
#
# FFmpeg configure script
#
# Copyright (c) 2000-2002 Fabrice Bellard
# Copyright (c) 2005-2008 Diego Biurrun
# Copyright (c) 2005-2008 Mans Rullgard
#
# 注释：雷霄骅
# leixiaohua1020@126.com
# http://blog.csdn.net/leixiaohua1020
#
# 添加了注释的FFmpeg的Configure文件
#
# Prevent locale nonsense from breaking basic text processing.
LC_ALL=C
export LC_ALL

# make sure we are running under a compatible shell
# try to make this part work with most shells

try_exec(){
    echo "Trying shell $1"
    type "$1" > /dev/null 2>&1 && exec "$@"
}
```

```
unset foo
(: ${foo%%bar}) 2> /dev/null
E1="$?"

(: ${foo?}) 2> /dev/null
E2="$?"

if test "$E1" != 0 || test "$E2" = 0; then
    echo "Broken shell detected.  Trying alternatives."
    export FF_CONF_EXEC
    if test "0$FF_CONF_EXEC" -lt 1; then
        FF_CONF_EXEC=1
        try_exec bash "$0" "$@"
    fi
    if test "0$FF_CONF_EXEC" -lt 2; then
        FF_CONF_EXEC=2
        try_exec ksh "$0" "$@"
    fi
    if test "0$FF_CONF_EXEC" -lt 3; then
        FF_CONF_EXEC=3
        try_exec /usr/xpg4/bin/sh "$0" "$@"
    fi
    echo "No compatible shell script interpreter found."
    echo "This configure script requires a POSIX-compatible shell"
    echo "such as bash or ksh."
    echo "THIS IS NOT A BUG IN FFMPEG, DO NOT REPORT IT AS SUCH."
    echo "Instead, install a working POSIX-compatible shell."
    echo "Disabling this configure test will create a broken FFmpeg."
    if test "$BASH_VERSION" = '2.04.0(1)-release'; then
        echo "This bash version ($BASH_VERSION) is broken on your platform."
        echo "Upgrade to a later version if available."
    fi
    exit 1
fi
#帮助菜单
show_help(){
cat <<EOF
Usage: configure [options]
Options: [defaults in brackets after descriptions]

Standard options:
  --help                   print this message
  --logfile=FILE           log tests and output to FILE [config.log]
  --disable-logging        do not log configure debug information
  --prefix=PREFIX          install in PREFIX [$prefix]
  --bindir=DIR             install binaries in DIR [PREFIX/bin]
  --datadir=DIR            install data files in DIR [PREFIX/share/ffmpeg]
  --libdir=DIR             install libs in DIR [PREFIX/lib]
  --shlibdir=DIR           install shared libs in DIR [PREFIX/lib]
  --incdir=DIR             install includes in DIR [PREFIX/include]
  --mandir=DIR             install man page in DIR [PREFIX/share/man]

Configuration options:
  --disable-static         do not build static libraries [no]
  --enable-shared          build shared libraries [no]
  --enable-gpl             allow use of GPL code, the resulting libs
                           and binaries will be under GPL [no]
  --enable-version3        upgrade (L)GPL to version 3 [no]
  --enable-nonfree         allow use of nonfree code, the resulting libs
                           and binaries will be unredistributable [no]
  --disable-doc            do not build documentation
  --disable-ffmpeg         disable ffmpeg build
  --disable-ffplay         disable ffplay build
  --disable-ffprobe        disable ffprobe build
  --disable-ffserver       disable ffserver build
  --disable-avdevice       disable libavdevice build
  --disable-avcodec        disable libavcodec build
  --disable-avformat       disable libavformat build
  --disable-swresample     disable libswresample build
  --disable-swscale        disable libswscale build
  --disable-postproc       disable libpostproc build
  --disable-avfilter       disable video filter support [no]
  --disable-pthreads       disable pthreads [auto]
  --disable-w32threads     disable Win32 threads [auto]
  --disable-os2threads     disable OS/2 threads [auto]
  --enable-x11grab         enable X11 grabbing [no]
  --disable-network        disable network support [no]
  --enable-gray            enable full grayscale support (slower color)
  --disable-swscale-alpha  disable alpha channel support in swscale
  --disable-fastdiv        disable table-based division
  --enable-small           optimize for size instead of speed
  --disable-aandct         disable AAN DCT code
  --disable-dct            disable DCT code
  --disable-fft            disable FFT code
  --disable-golomb         disable Golomb code
  --disable-huffman        disable Huffman code
  --disable-lpc            disable LPC code
  --disable-mdct           disable MDCT code
  --disable-rdft           disable RDFT code
  --enable-vaapi           enable VAAPI code [autodetect]
  --enable-vda             enable VDA code [autodetect]
```

```
  --enable-vda              enable VDA code [autodetect]
  --enable-vdpau            enable VDPAU code [autodetect]
  --disable-dxva2           disable DXVA2 code
  --disable-vda             disable VDA code
  --enable-runtime-cpudetect detect cpu capabilities at runtime (bigger binary)
  --enable-hardcoded-tables use hardcoded tables instead of runtime generation
  --disable-safe-bitstream-reader
                            disable buffer boundary checking in bitreaders
                            (faster, but may crash)
  --enable-memalign-hack    emulate memalign, interferes with memory debuggers
  --disable-everything      disable all components listed below
  --disable-encoder=NAME    disable encoder NAME
  --enable-encoder=NAME     enable encoder NAME
  --disable-encoders        disable all encoders
  --disable-decoder=NAME    disable decoder NAME
  --enable-decoder=NAME     enable decoder NAME
  --disable-decoders        disable all decoders
  --disable-hwaccel=NAME    disable hwaccel NAME
  --enable-hwaccel=NAME     enable hwaccel NAME
  --disable-hwaccels        disable all hwaccels
  --disable-muxer=NAME      disable muxer NAME
  --enable-muxer=NAME       enable muxer NAME
  --disable-muxers          disable all muxers
  --disable-demuxer=NAME    disable demuxer NAME
  --enable-demuxer=NAME     enable demuxer NAME
  --disable-demuxers        disable all demuxers
  --enable-parser=NAME      enable parser NAME
  --disable-parser=NAME     disable parser NAME
  --disable-parsers         disable all parsers
  --enable-bsf=NAME         enable bitstream filter NAME
  --disable-bsf=NAME        disable bitstream filter NAME
  --disable-bsfs            disable all bitstream filters
  --enable-protocol=NAME    enable protocol NAME
  --disable-protocol=NAME   disable protocol NAME
  --disable-protocols       disable all protocols
  --disable-indev=NAME      disable input device NAME
  --disable-outdev=NAME     disable output device NAME
  --disable-indevs          disable input devices
  --disable-outdevs         disable output devices
  --disable-devices         disable all devices
  --enable-filter=NAME      enable filter NAME
  --disable-filter=NAME     disable filter NAME
  --disable-filters         disable all filters
  --list-decoders           show all available decoders
  --list-encoders           show all available encoders
  --list-hwaccels           show all available hardware accelerators
  --list-muxers             show all available muxers
  --list-demuxers           show all available demuxers
  --list-parsers            show all available parsers
  --list-protocols          show all available protocols
  --list-bsfs               show all available bitstream filters
  --list-indevs             show all available input devices
  --list-outdevs            show all available output devices
  --list-filters            show all available filters

External library support:
  --enable-avisynth         enable reading of AVISynth script files [no]
  --enable-bzlib            enable bzlib [autodetect]
  --enable-frei0r           enable frei0r video filtering
  --enable-gnutls           enable gnutls [no]
  --enable-libaacplus       enable AAC+ encoding via libaacplus [no]
  --enable-libass           enable libass subtitles rendering [no]
  --enable-libcelt          enable CELT decoding via libcelt [no]
  --enable-libopencore-amrnb enable AMR-NB de/encoding via libopencore-amrnb [no]
  --enable-libopencore-amrwb enable AMR-WB decoding via libopencore-amrwb [no]
  --enable-libopencv        enable video filtering via libopencv [no]
  --enable-libcdio          enable audio CD grabbing with libcdio
  --enable-libdc1394        enable IIDC-1394 grabbing using libdc1394
                            and libraw1394 [no]
  --enable-libdirac         enable Dirac support via libdirac [no]
  --enable-libfaac          enable FAAC support via libfaac [no]
  --enable-libfreetype      enable libfreetype [no]
  --enable-libgsm           enable GSM support via libgsm [no]
  --enable-libmodplug       enable ModPlug via libmodplug [no]
  --enable-libmp3lame       enable MP3 encoding via libmp3lame [no]
  --enable-libnut           enable NUT (de)muxing via libnut,
                            native (de)muxer exists [no]
  --enable-libopenjpeg      enable JPEG 2000 encoding/decoding via OpenJPEG [no]
  --enable-libpulse         enable Pulseaudio input via libpulse [no]
  --enable-librtmp          enable RTMP[E] support via librtmp [no]
  --enable-libschroedinger enable Dirac support via libschroedinger [no]
  --enable-libspeex         enable Speex support via libspeex [no]
  --enable-libstagefright-h264  enable H.264 decoding via libstagefright [no]
  --enable-libtheora        enable Theora encoding via libtheora [no]
  --enable-libutvideo       enable Ut Video decoding via libutvideo [no]
  --enable-libv4l2          enable libv4l2/v4l-utils [no]
  --enable-libvo-aacenc     enable AAC encoding via libvo-aacenc [no]
  --enable-libvo-amrwbenc   enable AMR-WB encoding via libvo-amrwbenc [no]
  --enable-libvorbis        enable Vorbis encoding via libvorbis,
                            native implementation exists [no]
  --enable-libvpx           enable VP8 support via libvpx [no]
  --enable-libx264          enable H.264 encoding via x264 [no]
```

```
  --enable-libx264         enable H.264 encoding via x264 [no]
  --enable-libxavs         enable AVS encoding via xavs [no]
  --enable-libxvid         enable Xvid encoding via xvidcore,
                           native MPEG-4/Xvid encoder exists [no]
  --enable-openal          enable OpenAL 1.1 capture support [no]
  --enable-mlib            enable Sun medialib [no]
  --enable-openssl         enable openssl [no]
  --enable-zlib            enable zlib [autodetect]

Advanced options (experts only):
  --cross-prefix=PREFIX    use PREFIX for compilation tools [$cross_prefix]
  --enable-cross-compile   assume a cross-compiler is used
  --sysroot=PATH           root of cross-build tree
  --sysinclude=PATH        location of cross-build system headers
  --target-os=OS           compiler targets OS [$target_os]
  --target-exec=CMD        command to run executables on target
  --target-path=DIR        path to view of build directory on target
  --nm=NM                  use nm tool NM [$nm_default]
  --ar=AR                  use archive tool AR [$ar_default]
  --as=AS                  use assembler AS [$as_default]
  --yasmexe=EXE            use yasm-compatible assembler EXE [$yasmexe_default]
  --cc=CC                  use C compiler CC [$cc_default]
  --cxx=CXX                use C compiler CXX [$cxx_default]
  --ld=LD                  use linker LD [$ld_default]
  --host-cc=HOSTCC         use host C compiler HOSTCC
  --host-cflags=HCFLAGS    use HCFLAGS when compiling for host
  --host-ldflags=HLDFLAGS  use HLDFLAGS when linking for host
  --host-libs=HLIBS        use libs HLIBS when linking for host
  --extra-cflags=ECFLAGS   add ECFLAGS to CFLAGS [$CFLAGS]
  --extra-cxxflags=ECFLAGS add ECFLAGS to CXXFLAGS [$CXXFLAGS]
  --extra-ldflags=ELDFLAGS add ELDFLAGS to LDFLAGS [$LDFLAGS]
  --extra-libs=ELIBS       add ELIBS [$ELIBS]
  --extra-version=STRING   version string suffix []
  --build-suffix=SUFFIX    library name suffix []
  --progs-suffix=SUFFIX    program name suffix []
  --arch=ARCH              select architecture [$arch]
  --cpu=CPU                select the minimum required CPU (affects
                           instruction selection, may crash on older CPUs)
  --disable-asm            disable all assembler optimizations
  --disable-altivec        disable AltiVec optimizations
  --disable-amd3dnow       disable 3DNow! optimizations
  --disable-amd3dnowext    disable 3DNow! extended optimizations
  --disable-mmx            disable MMX optimizations
  --disable-mmx2           disable MMX2 optimizations
  --disable-sse            disable SSE optimizations
  --disable-ssse3          disable SSSE3 optimizations
  --disable-avx            disable AVX optimizations
  --disable-armv5te        disable armv5te optimizations
  --disable-armv6          disable armv6 optimizations
  --disable-armv6t2        disable armv6t2 optimizations
  --disable-armvfp         disable ARM VFP optimizations
  --disable-iwmmxt         disable iwmmxt optimizations
  --disable-mmi            disable MMI optimizations
  --disable-neon           disable NEON optimizations
  --disable-vis            disable VIS optimizations
  --disable-yasm           disable use of yasm assembler
  --enable-pic             build position-independent code
  --malloc-prefix=PFX      prefix malloc and related names with PFX
  --enable-sram            allow use of on-chip SRAM
  --disable-symver         disable symbol versioning
  --optflags               override optimization-related compiler flags
  --postproc-version=V     build libpostproc version V.
                           Where V can be '$ALT_PP_VER_MAJOR.$ALT_PP_VER_MINOR.$ALT_PP_VER_MICRO' or 'current'. [$postproc_version_de
lt]

Developer options (useful when working on FFmpeg itself):
  --enable-coverage        build with test coverage instrumentation
  --disable-debug          disable debugging symbols
  --enable-debug=LEVEL     set the debug level [$debuglevel]
  --disable-optimizations  disable compiler optimizations
  --enable-extra-warnings  enable more compiler warnings
  --disable-stripping      disable stripping of executables and shared libraries
  --valgrind=VALGRIND      run "make fate" tests through valgrind to detect memory
                           leaks and errors, using the specified valgrind binary.
                           Cannot be combined with --target-exec
  --samples=PATH           location of test samples for FATE, if not set use
                           \$FATE_SAMPLES at make invocation time.

NOTE: Object files are built at the place where configure is launched.
EOF
  exit 0
}

quotes='"""'

#日志config.log
log(){
    echo "$@" >> $logfile
}

log_file(){
    log BEGIN $1
```

```
    pr -n -t $1 >> $logfile
    log END $1
}

echolog(){
    log "$@"
    echo "$@"
}

warn(){
    log "WARNING: $*"
    WARNINGS="${WARNINGS}WARNING: $*\n"
}

#出错了
die(){
    echolog "$@"
    cat <<EOF

If you think configure made a mistake, make sure you are using the latest
version from Git.  If the latest version fails, report the problem to the
ffmpeg-user@ffmpeg.org mailing list or IRC #ffmpeg on irc.freenode.net.
EOF
    if disabled logging; then
        cat <<EOF
Rerun configure with logging enabled (do not use --disable-logging), and
include the log this produces with your report.
EOF
    else
cat <<EOF
Include the log file "$logfile" produced by configure as this will help
solving the problem.
EOF
    fi
    exit 1
}

# Avoid locale weirdness, besides we really just want to translate ASCII.
toupper(){
    echo "$@" | tr abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ
}

tolower(){
    echo "$@" | tr ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz
}

c_escape(){
    echo "$*" | sed 's/["\\]/\\\0/g'
}

sh_quote(){
    v=$(echo "$1" | sed "s/'/'\\\\''/g")
    test "x$v" = "x${v#*[!A-Za-z0-9_/.+-]}" || v="'$v'"
    echo "$v"
}

cleanws(){
    echo "$@" | sed 's/^ *//;s/  */ /g;s/ *$//'
}

filter(){
    pat=$1
    shift
    for v; do
        eval "case $v in $pat) echo $v ;; esac"
    done
}

filter_out(){
    pat=$1
    shift
    for v; do
        eval "case $v in $pat) ;; *) echo $v ;; esac"
    done
}

map(){
    m=$1
    shift
    for v; do eval $m; done
}

#第一个参数为值，后面的参数为变量
set_all(){
    value=$1
    shift
    for var in $*; do
        eval $var=$value
    done
}

```

```
set_weak(){
    value=$1
    shift
    for var; do
        eval : \${$var:=$value}
    done
}

set_safe(){
    var=$1
    shift
    eval $(echo "$var" | sed 's/[^A-Za-z0-9_]/_/g')='$*'
}

get_safe(){
    eval echo \$$(echo "$1" | sed 's/[^A-Za-z0-9_]/_/g')
}

pushvar(){
    for var in $*; do
        eval level=\${${var}_level:=0}
        eval ${var}_${level}="\$$var"
        eval ${var}_level=$(($level+1))
    done
}

popvar(){
    for var in $*; do
        eval level=\${${var}_level:-0}
        test $level = 0 && continue
        eval level=$(($level-1))
        eval $var="\${${var}_${level}}"
        eval ${var}_level=$level
        eval unset ${var}_${level}
    done
}
#把所有输入参数的值设置为"yes"
enable(){
    set_all yes $*
}
#把所有输入参数的值设置为"no"
disable(){
    set_all no $*
}

enable_weak(){
    set_weak yes $*
}

disable_weak(){
    set_weak no $*
}

enable_safe(){
    for var; do
        enable $(echo "$var" | sed 's/[^A-Za-z0-9_]/_/g')
    done
}

disable_safe(){
    for var; do
        disable $(echo "$var" | sed 's/[^A-Za-z0-9_]/_/g')
    done
}

do_enable_deep(){
    for var; do
        enabled $var && continue
        eval sel="\$${var}_select"
        eval sgs="\$${var}_suggest"
        pushvar var sgs
        enable_deep $sel
        popvar sgs
        enable_deep_weak $sgs
        popvar var
    done
}

enable_deep(){
    do_enable_deep $*
    enable $*
}

enable_deep_weak(){
    do_enable_deep $*
    enable_weak $*
}
#是否开启该组件
#主要通过看该组件的取值为"yes"还是"no"
#示例 enabled yasm
enabled(){
```

```
# "#"代表删除从前往后最小匹配的内容
# 在这里，即删除"!"，去掉${1}的"!"?
# 这句话的意思可能是，检查去掉"!"之后，${1}是否相等？即检查${1}中是否包含"!"
# PS：一般很少有包含"!"的情况吧 = =!
# 如果包含"!"，op取值为"="，否则取值"!="
    test "${1#!}" = "$1" && op== || op=!=
# 进行比较，看看取值是否为"yes"
# 为什么要在两边的前面都加一个"x"?
    eval test "x\$${1#!}" $op "xyes"
}
#是否关闭该组件
#和enabled相对应
disabled(){
    test "${1#!}" = "$1" && op== || op=!=
    #看看取值是否为"no"
    eval test "x\$${1#!}" $op "xno"
}

enabled_all(){
    for opt; do
        enabled $opt || return 1
    done
}

disabled_all(){
    for opt; do
        disabled $opt || return 1
    done
}

enabled_any(){
    for opt; do
        enabled $opt && return 0
    done
}

disabled_any(){
    for opt; do
        disabled $opt && return 0
    done
    return 1
}
#设置默认值
set_default(){
    for opt; do
        eval : \${$opt:=\$${opt}_default}
    done
}

is_in(){
    value=$1
    shift
    for var in $*; do
        [ $var = $value ] && return 0
    done
    return 1
}

check_deps(){
    for cfg; do
        cfg="${cfg#!}"
        enabled ${cfg}_checking && die "Circular dependency for $cfg."
        disabled ${cfg}_checking && continue
        enable ${cfg}_checking

        eval dep_all="\$${cfg}_deps"
        eval dep_any="\$${cfg}_deps_any"
        eval dep_sel="\$${cfg}_select"
        eval dep_sgs="\$${cfg}_suggest"
        eval dep_ifa="\$${cfg}_if"
        eval dep_ifn="\$${cfg}_if_any"

        pushvar cfg dep_all dep_any dep_sel dep_sgs dep_ifa dep_ifn
        check_deps $dep_all $dep_any $dep_sel $dep_sgs $dep_ifa $dep_ifn
        popvar cfg dep_all dep_any dep_sel dep_sgs dep_ifa dep_ifn

        [ -n "$dep_ifa" ] && { enabled_all $dep_ifa && enable_weak $cfg; }
        [ -n "$dep_ifn" ] && { enabled_any $dep_ifn && enable_weak $cfg; }
        enabled_all  $dep_all || disable $cfg
        enabled_any  $dep_any || disable $cfg
        disabled_any $dep_sel && disable $cfg

        if enabled $cfg; then
            eval dep_extralibs="\$${cfg}_extralibs"
            test -n "$dep_extralibs" && add_extralibs $dep_extralibs
            enable_deep $dep_sel
            enable_deep_weak $dep_sgs
        fi

        disable ${cfg}_checking
    done
```

```
}
#输出config.h的时候使用
#调用示例：print_config_h ffplay CONFIG_FFPLAY
print_config_h(){
#command1  && command2
#&&左边的命令（命令1）返回真(即返回0，成功被执行）后，&&右边的命令（命令2）才能够被执行
#command1 || command2
#||左边的命令（命令1）未执行成功，那么就执行||右边的命令（命令2）
    enabled $1 && v=1 || v=0
#示例：#define CONFIG_FFPLAY 1
    echo "#define $2 $v"
}
#输出config.mak的时候使用
print_config_mak(){
    enabled $1 && v= || v=!
    echo "$v$2=yes"
}
# 输出config.asm的时候使用
print_config_asm(){
    enabled $1 && echo "%define $2"
}
# 输出文本到config.mak，config.h等文件
# 该函数的示例调用方法：print_config CONFIG_ "$config_files" $CONFIG_LIST
print_config(){
# 前缀
    pfx=$1
# 文件列表
    files=$2
    # 位置参数可以用shift命令左移。比如shift 3表示原来的$4现在变成$1
    shift 2
    #for循环中，当没有in指定列表时，for会默认取命令行参数列表。
    #在这里取的就是$CONFIG_LIST 等
    for cfg; do
    # toupper()：转换为大写
        ucname="$(toupper $cfg)"
        # files= config.h config.mak config.asm
        # 循环输出
        for f in $files; do
        # "x#*/"代表去取x的第一个slash之后的所有内容（不包括slash）
        # "#"代表删除从前往后最小匹配的内容
        # "f##*."代表去取f的第一个"."之后的所有内容。在这里是“h”、“mak”等
        # 在这里print_config_h(),print_config_mak(),print_config_asm()
            "print_config_${f##*.}" $cfg ${pfx}${ucname} >>$f
        done
    done
}

print_enabled(){
    test "$1" = -n && end=" " && shift || end="\n"
    suf=$1
    shift
    for v; do
        enabled $v && printf "%s$end" ${v%$suf};
    done
}
# 添加
# 示例append config_files "config.asm"
append(){
    var=$1
    shift
    # eval命令将会首先扫描命令行进行所有的置换，然后再执行该命令
    # 按照上面的示例，置换后为 config_files="$config_files config.asm"
    eval "$var=\"\$$var $*\""
}

prepend(){
    var=$1
    shift
    eval "$var=\"$* \$$var\""
}

add_cppflags(){
    append CPPFLAGS $($filter_cppflags "$@")
}

add_cflags(){
    append CFLAGS $($filter_cflags "$@")
}

add_cxxflags(){
    append CXXFLAGS $($filter_cflags "$@")
}

add_asflags(){
    append ASFLAGS $($filter_asflags "$@")
}

add_ldflags(){
    append LDFLAGS "$@"
}

add_extralibs(){
```

```
663.  add_extralibs(){
664.      prepend extralibs "$@"
665.  }
666.  #2> 代表的是错误输出的重定向
667.  #标准的输入、输出、和错误输出分别表示STDIN STDOUT STDERR 也可以用数字表示 0  1  2
668.  #在这里也就是标准输出STDOUT 和 标准错误输出STDERR 都输入到了$logfile文件
669.  check_cmd(){
670.      log "$@"
671.      "$@" >> $logfile 2>&1
672.  }
673.  #检查CC编译器
674.  check_cc(){
675.      log check_cc "$@"
676.      cat > $TMPC
677.      log_file $TMPC
678.      #很多检查都调用了这个check_cmd
679.      #-c 只编译不链接
680.      check_cmd $cc $CPPFLAGS $CFLAGS "$@" -c -o $TMPO $TMPC
681.  }
682.
683.  check_cxx(){
684.      log check_cxx "$@"
685.      cat > $TMPCPP
686.      log_file $TMPCPP
687.      check_cmd $cxx $CPPFLAGS $CFLAGS $CXXFLAGS "$@" -c -o $TMPO $TMPCPP
688.  }
689.
690.  check_cpp(){
691.      log check_cpp "$@"
692.      cat > $TMPC
693.      log_file $TMPC
694.      #-E选项，可以让编译器在预处理后停止，并输出预处理结果。
695.      check_cmd $cc $CPPFLAGS $CFLAGS "$@" -E -o $TMPO $TMPC
696.  }
697.
698.  check_as(){
699.      log check_as "$@"
700.      cat > $TMPC
701.      log_file $TMPC
702.      check_cmd $as $CPPFLAGS $ASFLAGS "$@" -c -o $TMPO $TMPC
703.  }
704.
705.  check_asm(){
706.      log check_asm "$@"
707.      name="$1"
708.      code="$2"
709.      shift 2
710.      disable $name
711.      check_as "$@" <<EOF && enable $name
712.  void foo(void){ __asm__ volatile($code); }
713.  EOF
714.  }
715.
716.  check_yasm(){
717.      log check_yasm "$@"
718.      echo "$1" > $TMPS
719.      log_file $TMPS
720.      shift 1
721.      check_cmd $yasmexe $YASMFLAGS "$@" -o $TMPO $TMPS
722.  }
723.
724.  check_ld(){
725.      log check_ld "$@"
726.      type=$1
727.      shift 1
728.      flags=''
729.      libs=''
730.      for f; do
731.          test "${f}" = "${f#-l}" && flags="$flags $f" || libs="$libs $f"
732.      done
733.      check_$type $($filter_cflags $flags) || return
734.      #编译连接
735.      check_cmd $ld $LDFLAGS $flags -o $TMPE $TMPO $libs $extralibs
736.  }
737.
738.  check_cppflags(){
739.      log check_cppflags "$@"
740.      set -- $($filter_cppflags "$@")
741.      check_cc "$@" <<EOF && append CPPFLAGS "$@"
742.  int x;
743.  EOF
744.  }
745.
746.  check_cflags(){
747.      log check_cflags "$@"
748.      set -- $($filter_cflags "$@")
749.      check_cc "$@" <<EOF && append CFLAGS "$@"
750.  int x;
751.  EOF
752.  }
753.
754.  check_cxxflags(){
```

```
check_cxxflags(){
    log check_cxxflags "$@"
    set -- $($filter_cflags "$@")
    check_cxx "$@" <<EOF && append CXXFLAGS "$@"
int x;
EOF
}

test_ldflags(){
    log test_ldflags "$@"
    check_ld "cc" "$@" <<EOF
int main(void){ return 0; }
EOF
}

check_ldflags(){
    log check_ldflags "$@"
    test_ldflags "$@" && add_ldflags "$@"
}
#检查头文件
#生成一个简单的源代码文件
check_header(){
    log check_header "$@"
    header=$1
    shift
    disable_safe $header
    check_cpp "$@" <<EOF && enable_safe $header
#include <$header>
int x;
EOF
}

check_func(){
    log check_func "$@"
    func=$1
    shift
    disable $func
    check_ld "cc" "$@" <<EOF && enable $func
extern int $func();
int main(void){ $func(); }
EOF
}
#检查数学函数
check_mathfunc(){
    log check_mathfunc "$@"
    #数学函数名称
    func=$1
    shift
    disable $func
    check_ld "cc" "$@" <<EOF && enable $func
#include <math.h>
float foo(float f) { return $func(f); }
int main(void){ return (int) foo; }
EOF
}

check_func_headers(){
    log check_func_headers "$@"
    headers=$1
    funcs=$2
    shift 2
    {
        for hdr in $headers; do
            echo "#include <$hdr>"
        done
        for func in $funcs; do
            echo "long check_$func(void) { return (long) $func; }"
        done
        echo "int main(void) { return 0; }"
    } | check_ld "cc" "$@" && enable $funcs && enable_safe $headers
}

check_class_headers_cpp(){
    log check_class_headers_cpp "$@"
    headers=$1
    classes=$2
    shift 2
    {
        for hdr in $headers; do
            echo "#include <$hdr>"
        done
        echo "int main(void) { "
        i=1
        for class in $classes; do
            echo "$class obj$i;"
            i=$(expr $i + 1)
        done
        echo "return 0; }"
    } | check_ld "cxx" "$@" && enable $funcs && enable_safe $headers
}

check_cpp_condition(){
```

```
    log check_cpp_condition "$@"
    header=$1
    condition=$2
    shift 2
    check_cpp $($filter_cppflags "$@") <<EOF
#include <$header>
#if !($condition)
#error "unsatisfied condition: $condition"
#endif
EOF
}
#检查类库
check_lib(){
    log check_lib "$@"
    header="$1"
    func="$2"
    shift 2
    check_header $header && check_func $func "$@" && add_extralibs "$@"
}

check_lib2(){
    log check_lib2 "$@"
    headers="$1"
    funcs="$2"
    shift 2
    check_func_headers "$headers" "$funcs" "$@" && add_extralibs "$@"
}

check_lib_cpp(){
    log check_lib_cpp "$@"
    headers="$1"
    classes="$2"
    shift 2
    check_class_headers_cpp "$headers" "$classes" "$@" && add_extralibs "$@"
}

check_pkg_config(){
    log check_pkg_config "$@"
    pkg="$1"
    headers="$2"
    funcs="$3"
    shift 3
    $pkg_config --exists $pkg 2>/dev/null || return
    pkg_cflags=$($pkg_config --cflags $pkg)
    pkg_libs=$($pkg_config --libs $pkg)
    check_func_headers "$headers" "$funcs" $pkg_cflags $pkg_libs "$@" &&
        set_safe ${pkg}_cflags $pkg_cflags   &&
        set_safe ${pkg}_libs   $pkg_libs
}

check_exec(){
    check_ld "cc" "$@" && { enabled cross_compile || $TMPE >> $logfile 2>&1; }
}

check_exec_crash(){
    code=$(cat)

    # exit() is not async signal safe.  _Exit (C99) and _exit (POSIX)
    # are safe but may not be available everywhere.  Thus we use
    # raise(SIGTERM) instead.  The check is run in a subshell so we
    # can redirect the "Terminated" message from the shell.  SIGBUS
    # is not defined by standard C so it is used conditionally.

    (check_exec "$@") >> $logfile 2>&1 <<EOF
#include <signal.h>
static void sighandler(int sig){
    raise(SIGTERM);
}
int func(void){
    $code
}
int main(void){
    signal(SIGILL, sighandler);
    signal(SIGFPE, sighandler);
    signal(SIGSEGV, sighandler);
#ifdef SIGBUS
    signal(SIGBUS, sighandler);
#endif
    return func();
}
EOF
}

check_type(){
    log check_type "$@"
    headers=$1
    type=$2
    shift 2
    disable_safe "$type"
    incs=""
    for hdr in $headers; do
```

```
        incs="$incs
#include <$hdr>"
    done
    check_cc "$@" <<EOF && enable_safe "$type"
$incs
$type v;
EOF
}

check_struct(){
    log check_type "$@"
    headers=$1
    struct=$2
    member=$3
    shift 3
    disable_safe "${struct}_${member}"
    incs=""
    for hdr in $headers; do
        incs="$incs
#include <$hdr>"
    done
    check_cc "$@" <<EOF && enable_safe "${struct}_${member}"
$incs
const void *p = &(($struct *)0)->$member;
EOF
}
#检查依赖项的时候使用
require(){
    name="$1"
    header="$2"
    func="$3"
    shift 3
    check_lib $header $func "$@" || die "ERROR: $name not found"
}

require2(){
    name="$1"
    headers="$2"
    func="$3"
    shift 3
    check_lib2 "$headers" $func "$@" || die "ERROR: $name not found"
}

require_cpp(){
    name="$1"
    headers="$2"
    classes="$3"
    shift 3
    check_lib_cpp "$headers" "$classes" "$@" || die "ERROR: $name not found"
}

require_pkg_config(){
    pkg="$1"
    check_pkg_config "$@" || die "ERROR: $pkg not found"
    add_cflags    $(get_safe ${pkg}_cflags)
    add_extralibs $(get_safe ${pkg}_libs)
}

check_host_cc(){
    log check_host_cc "$@"
    cat > $TMPC
    log_file $TMPC
    check_cmd $host_cc $host_cflags "$@" -c -o $TMPO $TMPC
}

check_host_cflags(){
    log check_host_cflags "$@"
    check_host_cc "$@" <<EOF && append host_cflags "$@"
int x;
EOF
}

apply(){
    file=$1
    shift
    "$@" < "$file" > "$file.tmp" && mv "$file.tmp" "$file" || rm "$file.tmp"
}
#比较两个文件${1}和${2}，如果两个文件发生了变化，则将${1}强制覆盖${2}
#该函数主要用于生成config.h
cp_if_changed(){
#cmp是二进制文件比较命令
#-s：只返回退出值。值0（真）指示相同的文件；值1（假）指示不同的文件；值 2 指示不可访问的文件或缺少选项。
    cmp -s "$1" "$2" && echo "$2 is unchanged" && return
    mkdir -p "$(dirname $2)"
    cp -f "$1" "$2"
}

# CONFIG_LIST contains configurable options, while HAVE_LIST is for
# system-dependent things.
#各种List
COMPONENT_LIST="
```

```
    bsfs
    decoders
    demuxers
    encoders
    filters
    hwaccels
    indevs
    muxers
    outdevs
    parsers
    protocols
"

PROGRAM_LIST="
    ffplay
    ffprobe
    ffserver
    ffmpeg
"

CONFIG_LIST="
#组件
    $COMPONENT_LIST
#可执行程序
    $PROGRAM_LIST
    avplay
    avprobe
    avserver
    aandct
    ac3dsp
    avcodec
    avdevice
    avfilter
    avformat
    avisynth
    bzlib
    crystalhd
    dct
    doc
    dwt
    dxva2
    fastdiv
    fft
    frei0r
    gnutls
    golomb
    gpl
    gray
    h264chroma
    h264dsp
    h264pred
    hardcoded_tables
    huffman
    libaacplus
    libass
    libcdio
    libcelt
    libdc1394
    libdirac
    libfaac
    libfreetype
    libgsm
    libmodplug
    libmp3lame
    libnut
    libopencore_amrnb
    libopencore_amrwb
    libopencv
    libopenjpeg
    libpulse
    librtmp
    libschroedinger
    libspeex
    libstagefright_h264
    libtheora
    libutvideo
    libv4l2
    libvo_aacenc
    libvo_amrwbenc
    libvorbis
    libvpx
    libx264
    libxavs
    libxvid
    lpc
    lsp
    mdct
    memalign_hack
    mlib
    mpegaudiodsp
    network
```

```
    nonfree
    openal
    openssl
    pic
    postproc
    rdft
    rtpdec
    runtime_cpudetect
    safe_bitstream_reader
    shared
    sinewin
    small
    sram
    static
    swresample
    swscale
    swscale_alpha
    thumb
    vaapi
    vda
    vdpau
    version3
    x11grab
    zlib
"

THREADS_LIST='
    pthreads
    w32threads
    os2threads
'

ARCH_LIST='
    alpha
    arm
    avr32
    avr32_ap
    avr32_uc
    bfin
    ia64
    m68k
    mips
    mips64
    parisc
    ppc
    ppc64
    s390
    sh4
    sparc
    sparc64
    tomi
    x86
    x86_32
    x86_64
'

ARCH_EXT_LIST='
    altivec
    amd3dnow
    amd3dnowext
    armv5te
    armv6
    armv6t2
    armvfp
    avx
    iwmmxt
    mmi
    mmx
    mmx2
    neon
    ppc4xx
    sse
    ssse3
    vfpv3
    vis
'

HAVE_LIST_PUB='
    bigendian
    fast_unaligned
'

HAVE_LIST="
    $ARCH_EXT_LIST
    $HAVE_LIST_PUB
    $THREADS_LIST
    aligned_stack
    alsa_asoundlib_h
    altivec_h
    arpa_inet_h
    asm_mod_y
    asm_types_h
```

```
asm_types_h
attribute_may_alias
attribute_packed
cbrtf
closesocket
cmov
dcbzl
dev_bktr_ioctl_bt848_h
dev_bktr_ioctl_meteor_h
dev_ic_bt8xx_h
dev_video_bktr_ioctl_bt848_h
dev_video_meteor_ioctl_meteor_h
dlfcn_h
dlopen
dos_paths
ebp_available
ebx_available
exp2
exp2f
fast_64bit
fast_clz
fast_cmov
fcntl
fork
getaddrinfo
gethrtime
GetProcessAffinityMask
GetProcessMemoryInfo
GetProcessTimes
getrusage
gnu_as
ibm_asm
inet_aton
inline_asm
isatty
kbhit
ldbrx
llrint
llrintf
local_aligned_16
local_aligned_8
localtime_r
log2
log2f
loongson
lrint
lrintf
lzo1x_999_compress
machine_ioctl_bt848_h
machine_ioctl_meteor_h
makeinfo
malloc_h
MapViewOfFile
memalign
mkstemp
mmap
PeekNamedPipe
poll_h
posix_memalign
round
roundf
sched_getaffinity
sdl
sdl_video_size
setmode
setrlimit
sndio_h
socklen_t
soundcard_h
strerror_r
strptime
struct_addrinfo
struct_ipv6_mreq
struct_rusage_ru_maxrss
struct_sockaddr_in6
struct_sockaddr_sa_len
struct_sockaddr_storage
struct_v4l2_frmivalenum_discrete
symver
symver_asm_label
symver_gnu_asm
sysconf
sysctl
sys_mman_h
sys_param_h
sys_resource_h
sys_select_h
sys_soundcard_h
sys_videoio_h
termios_h
threads
trunc
```

```
    truncf
    vfp_args
    VirtualAlloc
    winsock2_h
    xform_asm
    xmm_clobbers
    yasm
"

# options emitted with CONFIG_ prefix but not available on command line
CONFIG_EXTRA="
    avutil
    gplv3
    lgplv3
"

CMDLINE_SELECT="
    $ARCH_EXT_LIST
    $CONFIG_LIST
    $THREADS_LIST
    asm
    coverage
    cross_compile
    debug
    extra_warnings
    logging
    optimizations
    stripping
    symver
    yasm
"

PATHS_LIST='
    bindir
    datadir
    incdir
    libdir
    mandir
    prefix
    shlibdir
'

CMDLINE_SET="
    $PATHS_LIST
    ar
    arch
    as
    build_suffix
    progs_suffix
    cc
    cpu
    cross_prefix
    cxx
    dep_cc
    extra_version
    host_cc
    host_cflags
    host_ldflags
    host_libs
    host_os
    install
    ld
    logfile
    malloc_prefix
    nm
    optflags
    pkg_config
    samples
    strip
    sysinclude
    sysroot
    target_exec
    target_os
    target_path
    postproc_version
    valgrind
    yasmexe
"

CMDLINE_APPEND="
    extra_cflags
    extra_cxxflags
"

# code dependency declarations

# architecture extensions

armv5te_deps="arm"
armv6_deps="arm"
armv6t2_deps="arm"
```

```
armvfp_deps="arm"
iwmmxt_deps="arm"
neon_deps="arm"
vfpv3_deps="armvfp"

mmi_deps="mips"

altivec_deps="ppc"
ppc4xx_deps="ppc"

vis_deps="sparc"

x86_64_suggest="cmov fast_cmov"
amd3dnow_deps="mmx"
amd3dnowext_deps="amd3dnow"
mmx_deps="x86"
mmx2_deps="mmx"
sse_deps="mmx"
ssse3_deps="sse"
avx_deps="ssse3"

aligned_stack_if_any="ppc x86"
fast_64bit_if_any="alpha ia64 mips64 parisc64 ppc64 sparc64 x86_64"
fast_clz_if_any="alpha armv5te avr32 mips ppc x86"
fast_unaligned_if_any="armv6 ppc x86"

inline_asm_deps="!tms470"
need_memalign="altivec neon sse"

symver_if_any="symver_asm_label symver_gnu_asm"

# subsystems
dct_select="rdft"
mdct_select="fft"
rdft_select="fft"
mpegaudiodsp_select="dct"

# decoders / encoders / hardware accelerators
aac_decoder_select="mdct sinewin"
aac_encoder_select="mdct sinewin"
aac_latm_decoder_select="aac_decoder aac_latm_parser"
ac3_decoder_select="mdct ac3dsp ac3_parser"
ac3_encoder_select="mdct ac3dsp"
ac3_fixed_encoder_select="mdct ac3dsp"
alac_encoder_select="lpc"
amrnb_decoder_select="lsp"
amrwb_decoder_select="lsp"
atrac1_decoder_select="mdct sinewin"
atrac3_decoder_select="mdct"
binkaudio_dct_decoder_select="mdct rdft dct sinewin"
binkaudio_rdft_decoder_select="mdct rdft sinewin"
cavs_decoder_select="golomb"
cook_decoder_select="mdct sinewin"
cscd_decoder_suggest="zlib"
dca_decoder_select="mdct"
dnxhd_encoder_select="aandct"
dxa_decoder_select="zlib"
eac3_decoder_select="ac3_decoder"
eac3_encoder_select="mdct ac3dsp"
eamad_decoder_select="aandct"
eatgq_decoder_select="aandct"
eatqi_decoder_select="aandct"
ffv1_decoder_select="golomb"
flac_decoder_select="golomb"
flac_encoder_select="golomb lpc"
flashsv_decoder_select="zlib"
flashsv_encoder_select="zlib"
flashsv2_encoder_select="zlib"
flashsv2_decoder_select="zlib"
flv_decoder_select="h263_decoder"
flv_encoder_select="h263_encoder"
fraps_decoder_select="huffman"
h261_encoder_select="aandct"
h263_decoder_select="h263_parser"
h263_encoder_select="aandct"
h263_vaapi_hwaccel_select="vaapi h263_decoder"
h263i_decoder_select="h263_decoder"
h263p_encoder_select="h263_encoder"
h264_crystalhd_decoder_select="crystalhd h264_mp4toannexb_bsf h264_parser"
h264_decoder_select="golomb h264chroma h264dsp h264pred"
h264_dxva2_hwaccel_deps="dxva2api_h"
h264_dxva2_hwaccel_select="dxva2 h264_decoder"
h264_vaapi_hwaccel_select="vaapi h264_decoder"
h264_vda_hwaccel_deps="VideoDecodeAcceleration_VDADecoder_h pthreads"
h264_vda_hwaccel_select="vda h264_decoder"
h264_vdpau_decoder_select="vdpau h264_decoder"
imc_decoder_select="fft mdct sinewin"
jpegls_decoder_select="golomb"
jpegls_encoder_select="golomb"
ljpeg_encoder_select="aandct"
loco_decoder_select="golomb"
```

```
mjpeg_encoder_select="aandct"
mlp_decoder_select="mlp_parser"
mp1_decoder_select="mpegaudiodsp"
mp1float_decoder_select="mpegaudiodsp"
mp2_decoder_select="mpegaudiodsp"
mp2float_decoder_select="mpegaudiodsp"
mp3_decoder_select="mpegaudiodsp"
mp3adu_decoder_select="mpegaudiodsp"
mp3adufloat_decoder_select="mpegaudiodsp"
mp3float_decoder_select="mpegaudiodsp"
mp3on4_decoder_select="mpegaudiodsp"
mp3on4float_decoder_select="mpegaudiodsp"
mpc7_decoder_select="mpegaudiodsp"
mpc8_decoder_select="mpegaudiodsp"
mpeg_vdpau_decoder_select="vdpau mpegvideo_decoder"
mpeg_xvmc_decoder_deps="X11_extensions_XvMClib_h"
mpeg_xvmc_decoder_select="mpegvideo_decoder"
mpeg1_vdpau_decoder_select="vdpau mpeg1video_decoder"
mpeg1_vdpau_hwaccel_select="vdpau mpeg1video_decoder"
mpeg1video_encoder_select="aandct"
mpeg2_crystalhd_decoder_select="crystalhd"
mpeg2_dxva2_hwaccel_deps="dxva2api_h"
mpeg2_dxva2_hwaccel_select="dxva2 mpeg2video_decoder"
mpeg2_vdpau_hwaccel_select="vdpau mpeg2video_decoder"
mpeg2_vaapi_hwaccel_select="vaapi mpeg2video_decoder"
mpeg2video_encoder_select="aandct"
mpeg4_crystalhd_decoder_select="crystalhd"
mpeg4_decoder_select="h263_decoder mpeg4video_parser"
mpeg4_encoder_select="h263_encoder"
mpeg4_vaapi_hwaccel_select="vaapi mpeg4_decoder"
mpeg4_vdpau_decoder_select="vdpau mpeg4_decoder"
msmpeg4_crystalhd_decoder_select="crystalhd"
msmpeg4v1_decoder_select="h263_decoder"
msmpeg4v1_encoder_select="h263_encoder"
msmpeg4v2_decoder_select="h263_decoder"
msmpeg4v2_encoder_select="h263_encoder"
msmpeg4v3_decoder_select="h263_decoder"
msmpeg4v3_encoder_select="h263_encoder"
nellymoser_decoder_select="mdct sinewin"
nellymoser_encoder_select="mdct sinewin"
png_decoder_select="zlib"
png_encoder_select="zlib"
qcelp_decoder_select="lsp"
qdm2_decoder_select="mdct rdft mpegaudiodsp"
ra_144_encoder_select="lpc"
rv10_decoder_select="h263_decoder"
rv10_encoder_select="h263_encoder"
rv20_decoder_select="h263_decoder"
rv20_encoder_select="h263_encoder"
rv30_decoder_select="golomb h264chroma h264pred"
rv40_decoder_select="golomb h264chroma h264pred"
shorten_decoder_select="golomb"
sipr_decoder_select="lsp"
snow_decoder_select="dwt"
snow_encoder_select="aandct dwt"
sonic_decoder_select="golomb"
sonic_encoder_select="golomb"
sonic_ls_encoder_select="golomb"
svq1_encoder_select="aandct"
svq3_decoder_select="golomb h264chroma h264dsp h264pred"
svq3_decoder_suggest="zlib"
theora_decoder_select="vp3_decoder"
tiff_decoder_suggest="zlib"
tiff_encoder_suggest="zlib"
truehd_decoder_select="mlp_decoder"
tscc_decoder_select="zlib"
twinvq_decoder_select="mdct lsp sinewin"
vc1_crystalhd_decoder_select="crystalhd"
vc1_decoder_select="h263_decoder h264chroma"
vc1_dxva2_hwaccel_deps="dxva2api_h"
vc1_dxva2_hwaccel_select="dxva2 vc1_decoder"
vc1_vaapi_hwaccel_select="vaapi vc1_decoder"
vc1_vdpau_decoder_select="vdpau vc1_decoder"
vc1image_decoder_select="vc1_decoder"
vorbis_decoder_select="mdct"
vorbis_encoder_select="mdct"
vp6_decoder_select="huffman"
vp6a_decoder_select="vp6_decoder"
vp6f_decoder_select="vp6_decoder"
vp8_decoder_select="h264pred"
wmapro_decoder_select="mdct sinewin"
wmav1_decoder_select="mdct sinewin"
wmav1_encoder_select="mdct sinewin"
wmav2_decoder_select="mdct sinewin"
wmav2_encoder_select="mdct sinewin"
wmavoice_decoder_select="lsp rdft dct mdct sinewin"
wmv1_decoder_select="h263_decoder"
wmv1_encoder_select="h263_encoder"
wmv2_decoder_select="h263_decoder"
wmv2_encoder_select="h263_encoder"
wmv3_decoder_select="vc1_decoder"
```

```
wmv3_crystalhd_decoder_select="crystalhd"
wmv3_dxva2_hwaccel_select="vc1_dxva2_hwaccel"
wmv3_vaapi_hwaccel_select="vc1_vaapi_hwaccel"
wmv3_vdpau_decoder_select="vc1_vdpau_decoder"
wmv3image_decoder_select="wmv3_decoder"
zlib_decoder_select="zlib"
zlib_encoder_select="zlib"
zmbv_decoder_select="zlib"
zmbv_encoder_select="zlib"

crystalhd_deps="libcrystalhd_libcrystalhd_if_h"
vaapi_deps="va_va_h"
vda_deps="VideoDecodeAcceleration_VDADecoder_h pthreads"
vdpau_deps="vdpau_vdpau_h vdpau_vdpau_x11_h"

# parsers
h264_parser_select="golomb h264chroma h264dsp h264pred"

# external libraries
libaacplus_encoder_deps="libaacplus"
libcelt_decoder_deps="libcelt"
libdirac_decoder_deps="libdirac !libschroedinger"
libdirac_encoder_deps="libdirac"
libfaac_encoder_deps="libfaac"
libgsm_decoder_deps="libgsm"
libgsm_encoder_deps="libgsm"
libgsm_ms_decoder_deps="libgsm"
libgsm_ms_encoder_deps="libgsm"
libmodplug_demuxer_deps="libmodplug"
libmp3lame_encoder_deps="libmp3lame"
libopencore_amrnb_decoder_deps="libopencore_amrnb"
libopencore_amrnb_encoder_deps="libopencore_amrnb"
libopencore_amrwb_decoder_deps="libopencore_amrwb"
libopenjpeg_decoder_deps="libopenjpeg"
libopenjpeg_encoder_deps="libopenjpeg"
libschroedinger_decoder_deps="libschroedinger"
libschroedinger_encoder_deps="libschroedinger"
libspeex_decoder_deps="libspeex"
libspeex_encoder_deps="libspeex"
libstagefright_h264_decoder_deps="libstagefright_h264"
libtheora_encoder_deps="libtheora"
libvo_aacenc_encoder_deps="libvo_aacenc"
libvo_amrwbenc_encoder_deps="libvo_amrwbenc"
libvorbis_encoder_deps="libvorbis"
libvpx_decoder_deps="libvpx"
libvpx_encoder_deps="libvpx"
libx264_encoder_deps="libx264"
libx264rgb_encoder_deps="libx264"
libxavs_encoder_deps="libxavs"
libxvid_encoder_deps="libxvid"
libutvideo_decoder_deps="libutvideo gpl"

# demuxers / muxers
ac3_demuxer_select="ac3_parser"
asf_stream_muxer_select="asf_muxer"
avisynth_demuxer_deps="avisynth"
dirac_demuxer_select="dirac_parser"
eac3_demuxer_select="ac3_parser"
flac_demuxer_select="flac_parser"
ipod_muxer_select="mov_muxer"
libnut_demuxer_deps="libnut"
libnut_muxer_deps="libnut"
matroska_audio_muxer_select="matroska_muxer"
matroska_demuxer_suggest="zlib bzlib"
mov_demuxer_suggest="zlib"
mp3_demuxer_select="mpegaudio_parser"
mp4_muxer_select="mov_muxer"
mpegtsraw_demuxer_select="mpegts_demuxer"
mxf_d10_muxer_select="mxf_muxer"
ogg_demuxer_select="golomb"
psp_muxer_select="mov_muxer"
rtp_demuxer_select="sdp_demuxer"
rtpdec_select="asf_demuxer rm_demuxer rtp_protocol mpegts_demuxer mov_demuxer"
rtsp_demuxer_select="http_protocol rtpdec"
rtsp_muxer_select="rtp_muxer http_protocol rtp_protocol"
sap_demuxer_select="sdp_demuxer"
sap_muxer_select="rtp_muxer rtp_protocol"
sdp_demuxer_select="rtpdec"
spdif_muxer_select="aac_parser"
tg2_muxer_select="mov_muxer"
tgp_muxer_select="mov_muxer"
w64_demuxer_deps="wav_demuxer"

# indevs / outdevs
alsa_indev_deps="alsa_asoundlib_h snd_pcm_htimestamp"
alsa_outdev_deps="alsa_asoundlib_h"
bktr_indev_deps_any="dev_bktr_ioctl_bt848_h machine_ioctl_bt848_h dev_video_bktr_ioctl_bt848_h dev_ic_bt8xx_h"
dshow_indev_deps="IBaseFilter"
dshow_indev_extralibs="-lpsapi -lole32 -lstrmiids -luuid"
dv1394_indev_deps="dv1394 dv_demuxer"
fbdev_indev_deps="linux_fb_h"
```

```
jack_indev_deps="jack_jack_h sem_timedwait"
lavfi_indev_deps="avfilter"
libcdio_indev_deps="libcdio"
libdc1394_indev_deps="libdc1394"
libv4l2_indev_deps="libv4l2"
openal_indev_deps="openal"
oss_indev_deps_any="soundcard_h sys_soundcard_h"
oss_outdev_deps_any="soundcard_h sys_soundcard_h"
pulse_indev_deps="libpulse"
sdl_outdev_deps="sdl"
sndio_indev_deps="sndio_h"
sndio_outdev_deps="sndio_h"
v4l_indev_deps="linux_videodev_h"
v4l2_indev_deps_any="linux_videodev2_h sys_videoio_h"
vfwcap_indev_deps="capCreateCaptureWindow vfwcap_defines"
vfwcap_indev_extralibs="-lavicap32"
x11_grab_device_indev_deps="x11grab XShmCreateImage"
x11_grab_device_indev_extralibs="-lX11 -lXext -lXfixes"

# protocols
gopher_protocol_deps="network"
httpproxy_protocol_deps="network"
httpproxy_protocol_select="tcp_protocol"
http_protocol_deps="network"
http_protocol_select="tcp_protocol"
https_protocol_select="tls_protocol"
mmsh_protocol_select="http_protocol"
mmst_protocol_deps="network"
rtmp_protocol_select="tcp_protocol"
rtp_protocol_select="udp_protocol"
tcp_protocol_deps="network"
tls_protocol_deps_any="openssl gnutls"
tls_protocol_select="tcp_protocol"
udp_protocol_deps="network"

# filters
amovie_filter_deps="avcodec avformat"
ass_filter_deps="libass"
blackframe_filter_deps="gpl"
boxblur_filter_deps="gpl"
cropdetect_filter_deps="gpl"
delogo_filter_deps="gpl"
drawtext_filter_deps="libfreetype"
frei0r_filter_deps="frei0r dlopen"
frei0r_src_filter_deps="frei0r dlopen"
hqdn3d_filter_deps="gpl"
movie_filter_deps="avcodec avformat"
mp_filter_deps="gpl avcodec"
mptestsrc_filter_deps="gpl"
negate_filter_deps="lut_filter"
ocv_filter_deps="libopencv"
pan_filter_deps="swresample"
scale_filter_deps="swscale"
tinterlace_filter_deps="gpl"
yadif_filter_deps="gpl"

# libraries
avdevice_deps="avcodec avformat"
avformat_deps="avcodec"
postproc_deps="gpl"

# programs
ffplay_deps="avcodec avformat swscale sdl"
ffplay_select="buffersink_filter rdft"
ffprobe_deps="avcodec avformat"
ffserver_deps="avformat ffm_muxer fork rtp_protocol rtsp_demuxer"
ffserver_extralibs='$ldl'
ffmpeg_deps="avcodec avformat swscale swresample"
ffmpeg_select="buffersink_filter"

doc_deps="texi2html"

# tests

test_deps(){
    suf1=$1
    suf2=$2
    shift 2
    for v; do
        dep=${v%=*}
        tests=${v#*=}
        for name in ${tests}; do
            append ${name}_test_deps ${dep}$suf1 ${dep}$suf2
        done
    done
}

mxf_d10_test_deps="avfilter"
seek_lavf_mxf_d10_test_deps="mxf_d10_test"

test_deps _encoder _decoder                                     \
    adpcm_ima_qt                                                \
```

```
    adpcm_ima_qt                                                    \
    adpcm_ima_wav                                                   \
    adpcm_ms                                                        \
    adpcm_swf                                                       \
    adpcm_yamaha=adpcm_yam                                          \
    alac                                                            \
    asv1                                                            \
    asv2                                                            \
    bmp                                                             \
    dnxhd="dnxhd_1080i dnxhd_720p dnxhd_720p_rd"                    \
    dvvideo="dv dv_411 dv50"                                        \
    ffv1                                                            \
    flac                                                            \
    flashsv                                                         \
    flv                                                             \
    adpcm_g726=g726                                                 \
    gif                                                             \
    h261                                                            \
    h263="h263 h263p"                                               \
    huffyuv                                                         \
    jpegls                                                          \
    mjpeg="jpg mjpeg ljpeg"                                         \
    mp2                                                             \
    mpeg1video="mpeg mpeg1b"                                        \
    mpeg2video="mpeg2 mpeg2_422 mpeg2_idct_int mpeg2_ilace mpeg2_ivlc_qprd" \
    mpeg2video="mpeg2thread mpeg2thread_ilace"                      \
    mpeg4="mpeg4 mpeg4_adap mpeg4_qpel mpeg4_qprd mpeg4adv mpeg4nr"     \
    mpeg4="mpeg4thread error rc"                                    \
    msmpeg4v3=msmpeg4                                               \
    msmpeg4v2                                                       \
    pbm=pbmpipe                                                     \
    pcx                                                             \
    pgm="pgm pgmpipe"                                               \
    png                                                             \
    ppm="ppm ppmpipe"                                               \
    rawvideo="rgb yuv"                                              \
    roq                                                             \
    rv10                                                            \
    rv20                                                            \
    sgi                                                             \
    snow="snow snowll"                                              \
    svq1                                                            \
    targa=tga                                                       \
    tiff                                                            \
    wmav1                                                           \
    wmav2                                                           \
    wmv1                                                            \
    wmv2                                                            \

test_deps _muxer _demuxer                                           \
    aiff                                                            \
    pcm_alaw=alaw                                                   \
    asf                                                             \
    au                                                              \
    avi                                                             \
    dv=dv_fmt                                                       \
    ffm                                                             \
    flv=flv_fmt                                                     \
    gxf                                                             \
    matroska=mkv                                                    \
    mmf                                                             \
    mov                                                             \
    pcm_mulaw=mulaw                                                 \
    mxf="mxf mxf_d10"                                               \
    nut                                                             \
    ogg                                                             \
    rawvideo=pixfmt                                                 \
    rm                                                              \
    swf                                                             \
    mpegts=ts                                                       \
    voc                                                             \
    wav                                                             \
    yuv4mpegpipe=yuv4mpeg                                           \

ac3_fixed_test_deps="ac3_fixed_encoder ac3_decoder rm_muxer rm_demuxer"
mpg_test_deps="mpeg1system_muxer mpegps_demuxer"

# 默认参数 default parameters
# 日志
logfile="config.log"

# 安装路径 installation paths
prefix_default="/usr/local"
bindir_default='${prefix}/bin'
datadir_default='${prefix}/share/ffmpeg'
incdir_default='${prefix}/include'
libdir_default='${prefix}/lib'
mandir_default='${prefix}/share/man'
shlibdir_default="$libdir_default"
postproc_version_default="current"

# 工具链 toolchain
```

```
# 工具链 toolchain
ar_default="ar"
cc_default="gcc"
cxx_default="g++"
cc_version=\"unknown\"
host_cc_default="gcc"
install="install"
ln_s="ln -sf"
nm_default="nm"
objformat="elf"
pkg_config_default=pkg-config
ranlib="ranlib"
strip_default="strip"
yasmexe_default="yasm"

nm_opts='-g'
nogas=":"

# 机器 machine
arch_default=$(uname -m)
cpu="generic"

# 操作系统 OS
target_os_default=$(tolower $(uname -s))
host_os=$target_os_default

# alternative libpostproc version
ALT_PP_VER_MAJOR=51
ALT_PP_VER_MINOR=2
ALT_PP_VER_MICRO=101
ALT_PP_VER=$ALT_PP_VER_MAJOR.$ALT_PP_VER_MINOR.$ALT_PP_VER_MICRO

# 选项 configurable options
# PROGRAM_LIST内容是 ffplay ffprobe ffserver ffmpeg
enable $PROGRAM_LIST

enable avcodec
enable avdevice
enable avfilter
enable avformat
enable avutil
enable postproc
enable stripping
enable swresample
enable swscale

enable asm
enable debug
enable doc
enable fastdiv
enable network
enable optimizations
enable safe_bitstream_reader
enable static
enable swscale_alpha

# 编译选项 build settings
SHFLAGS='-shared -Wl,-soname,$$(@F)'
FFSERVERLDFLAGS=-Wl,-E
# 前缀后缀
LIBPREF="lib"
LIBSUF=".a"
FULLNAME='$(NAME)$(BUILDSUF)'
# 名称
LIBNAME='$(LIBPREF)$(FULLNAME)$(LIBSUF)'
# 动态库前缀后缀
SLIBPREF="lib"
SLIBSUF=".so"
# 名称
SLIBNAME='$(SLIBPREF)$(FULLNAME)$(SLIBSUF)'
SLIBNAME_WITH_VERSION='$(SLIBNAME).$(LIBVERSION)'
SLIBNAME_WITH_MAJOR='$(SLIBNAME).$(LIBMAJOR)'
LIB_INSTALL_EXTRA_CMD='$$(RANLIB) "$(LIBDIR)/$(LIBNAME)"'
SLIB_INSTALL_NAME='$(SLIBNAME_WITH_VERSION)'
SLIB_INSTALL_LINKS='$(SLIBNAME_WITH_MAJOR) $(SLIBNAME)'

AS_O='-o $@'
CC_O='-o $@'
CXX_O='-o $@'

host_cflags='-D_ISOC99_SOURCE -O3 -g'
host_libs='-lm'

target_path='$(CURDIR)'

# since the object filename is not given with the -MM flag, the compiler
# is only able to print the basename, and we must add the path ourselves
DEPEND_CMD='$(DEPCC) $(DEPFLAGS) $< | sed -e "/^\#.*/d" -e "s,^[[:space:]]*$(*F)\\.o,$(@D)/$(*F).o," > $(@:.o=.d)'
DEPFLAGS='$(CPPFLAGS) $(CFLAGS) -MM'

# find source path
# $0就是该bash文件名
```

```
# dirname /home/lxh/test.txt 输出/home/lxh
if test -f configure; then
    source_path=.
else
    source_path=$(cd $(dirname "$0"); pwd)
    echo "$source_path" | grep -q '[[:blank:]]' &&
        die "Out of tree builds are impossible with whitespace in source path."
    test -e "$source_path/config.h" &&
        die "Out of tree builds are impossible with config.h in source dir."
fi
# 脚本名称叫test.sh
# 入参三个: 1 2 3
# 运行test.sh 1 2 3后
# $*为"1 2 3"（一起被引号包住）
# $@为"1" "2" "3"（分别被包住）
# $#为3（参数数量）
for v in "$@"; do
    r=${v#*=}
    l=${v%"$r"}
    r=$(sh_quote "$r")
    FFMPEG_CONFIGURATION="${FFMPEG_CONFIGURATION# } ${l}${r}"
done
# ${数字} 一般是位置参数的用法。
# 如果运行脚本的时候带参数，那么可以在脚本里通过 $1 获取第一个参数，$2 获取第二个参数
# 例如以ENCODER_LIST为例，$1为"encoder"，$2为"ENC"，$3为"libavcodec/allcodecs.c"
find_things(){
    thing=$1
    pattern=$2
    file=$source_path/$3
    # 处理一行字符串？挺复杂
    sed -n "s/^[^#]*$pattern.*([^,]*, *\([^,]*\)\(,.*\)*).*/\1_$thing/p" "$file"
}
#从allcodecs.c等文件中提取编解码器
ENCODER_LIST=$(find_things  encoder  ENC      libavcodec/allcodecs.c)
DECODER_LIST=$(find_things  decoder  DEC      libavcodec/allcodecs.c)
HWACCEL_LIST=$(find_things  hwaccel  HWACCEL  libavcodec/allcodecs.c)
PARSER_LIST=$(find_things   parser   PARSER   libavcodec/allcodecs.c)
BSF_LIST=$(find_things      bsf      BSF      libavcodec/allcodecs.c)
MUXER_LIST=$(find_things    muxer    _MUX     libavformat/allformats.c)
DEMUXER_LIST=$(find_things  demuxer  DEMUX    libavformat/allformats.c)
OUTDEV_LIST=$(find_things   outdev   OUTDEV   libavdevice/alldevices.c)
INDEV_LIST=$(find_things    indev    _IN      libavdevice/alldevices.c)
PROTOCOL_LIST=$(find_things protocol PROTOCOL libavformat/allformats.c)
FILTER_LIST=$(find_things   filter   FILTER   libavfilter/allfilters.c)

# 所有组件
ALL_COMPONENTS="
    $BSF_LIST
    $DECODER_LIST
    $DEMUXER_LIST
    $ENCODER_LIST
    $FILTER_LIST
    $HWACCEL_LIST
    $INDEV_LIST
    $MUXER_LIST
    $OUTDEV_LIST
    $PARSER_LIST
    $PROTOCOL_LIST
"

find_tests(){
    map "echo ${2}\${v}_test" $(ls "$source_path"/tests/ref/$1 | grep -v '[^-a-z0-9_]')
}

ACODEC_TESTS=$(find_tests acodec)
VCODEC_TESTS=$(find_tests vsynth1)
LAVF_TESTS=$(find_tests lavf)
LAVFI_TESTS=$(find_tests lavfi)
SEEK_TESTS=$(find_tests seek seek_)

ALL_TESTS="$ACODEC_TESTS $VCODEC_TESTS $LAVF_TESTS $LAVFI_TESTS $SEEK_TESTS"

pcm_test_deps=$(map 'echo ${v%_*}_decoder $v' $(filter pcm_* $ENCODER_LIST))

for n in $COMPONENT_LIST; do
    v=$(toupper ${n%s})_LIST
    eval enable \$$v
    eval ${n}_if_any="\$$v"
done

enable $ARCH_EXT_LIST $ALL_TESTS

die_unknown(){
    echo "Unknown option \"$1\"."
    echo "See $0 --help for available options."
    exit 1
}

show_list() {
    suffix=_$1
    shift
```

```
    echo $* | sed s/$suffix//g | tr ' ' '\n' | sort | pr -3 -t
    exit 0
}
# 解析各种各样的选项
#
# case分支语句的格式如下：
#   case $变量名 in
#       模式1）
#   命令序列1
#   ;;
#       模式2）
#   命令序列2
#   ;;
#       *）
#   默认执行的命令序列
#   ;;
#   esac
# case语句结构特点如下：
# case行尾必须为单词“in”，每一个模式必须以右括号“）”结束。
# 双分号“;;”表示命令序列结束。
# 最后的“*）”表示默认模式，当使用前面的各种模式均无法匹配该变量时，将执行“*）”后的命令序列。
#
#注意：opt不是参数列表（实际上也没有看见opt变量的定义）
#原因是处在for循环中，当你没有为in指定列表时，for会默认取命令行参数列表。
#因此“opt”这个名字实际上是可以随便取的
for opt do
# "#"用于去除特定字符前面的字符串
# optval内容为opt去掉"="以及其前面字符串之后的内容
    optval="${opt#*=}"
    case "$opt" in
    # 不同的选项
    --extra-ldflags=*) add_ldflags $optval
    ;;
    --extra-libs=*) add_extralibs $optval
    ;;
    --disable-devices) disable $INDEV_LIST $OUTDEV_LIST
    ;;
    --enable-debug=*) debuglevel="$optval"
    ;;
    --disable-everything)
    map 'eval unset \${$(toupper ${v%s})_LIST}' $COMPONENT_LIST
    ;;
    --enable-*=*|--disable-*=*)
    eval $(echo "${opt%%=*}" | sed 's/--/action=/;s/-/ thing=/')
    is_in "${thing}s" $COMPONENT_LIST || die_unknown "$opt"
    eval list=\$$(toupper $thing)_LIST
    name=$(echo "${optval}" | sed "s/,/_${thing}|/g")_${thing}
    $action $(filter "$name" $list)
    ;;
    --enable-?*|--disable-?*)
    eval $(echo "$opt" | sed 's/--/action=/;s/-/ option=/;s/-/_/g')
    if is_in $option $COMPONENT_LIST; then
        test $action = disable && action=unset
        eval $action \$$(toupper ${option%s})_LIST
    elif is_in $option $CMDLINE_SELECT; then
        $action $option
    else
        die_unknown $opt
    fi
    ;;
    --list-*)
        NAME="${opt#--list-}"
        is_in $NAME $COMPONENT_LIST || die_unknown $opt
        NAME=${NAME%s}
        eval show_list $NAME \$$(toupper $NAME)_LIST
    ;;
    --help|-h) show_help
    ;;
    *)
    #% 就是从右边开始删除符合条件的字符串（符合条件的最短字符串）
    #%%是删除符合条件的最长的字符串

    #删除“=”右边的内容
    optname="${opt%%=*}"
    #删除左边的“--”
    optname="${optname#--}"
    optname=$(echo "$optname" | sed 's/-/_/g')
    #看看是否在opt列表中，不在的话就会返回错误
    if is_in $optname $CMDLINE_SET; then
        eval $optname='$optval'
    elif is_in $optname $CMDLINE_APPEND; then
        append $optname "$optval"
    else
         die_unknown $opt
    fi
    ;;
    esac
done

disabled logging && logfile=/dev/null

```

```
echo "# $0 $FFMPEG_CONFIGURATION" > $logfile
set >> $logfile

test -n "$cross_prefix" && enable cross_compile

if enabled cross_compile; then
    test -n "$arch" && test -n "$target_os" ||
        die "Must specify target arch and OS when cross-compiling"
fi

set_default arch target_os postproc_version

# Check if we should build alternative libpostproc version instead of current
if   test "$postproc_version" = $ALT_PP_VER; then
  LIBPOSTPROC_VERSION=$ALT_PP_VER
  LIBPOSTPROC_VERSION_MAJOR=$ALT_PP_VER_MAJOR
  LIBPOSTPROC_VERSION_MINOR=$ALT_PP_VER_MINOR
  LIBPOSTPROC_VERSION_MICRO=$ALT_PP_VER_MICRO
elif test "$postproc_version" != current; then
  die "Invalid argument to --postproc-version. See --help output."
fi

ar_default="${cross_prefix}${ar_default}"
cc_default="${cross_prefix}${cc_default}"
cxx_default="${cross_prefix}${cxx_default}"
nm_default="${cross_prefix}${nm_default}"
pkg_config_default="${cross_prefix}${pkg_config_default}"
ranlib="${cross_prefix}${ranlib}"
strip_default="${cross_prefix}${strip_default}"

sysinclude_default="${sysroot}/usr/include"

set_default cc cxx nm pkg_config strip sysinclude yasmexe
enabled cross_compile || host_cc_default=$cc
set_default host_cc

if ! $pkg_config --version >/dev/null 2>&1; then
    warn "$pkg_config not found, library detection may fail."
    pkg_config=false
fi

exesuf() {
    case $1 in
        mingw32*|cygwin*|*-dos|freedos|opendos|os/2*|symbian) echo .exe ;;
    esac
}

EXESUF=$(exesuf $target_os)
HOSTEXESUF=$(exesuf $host_os)

# set temporary file name
: ${TMPDIR:=$TEMPDIR}
: ${TMPDIR:=$TMP}
: ${TMPDIR:=/tmp}

if ! check_cmd mktemp -u XXXXXX; then
    # simple replacement for missing mktemp
    # NOT SAFE FOR GENERAL USE
    mktemp(){
        echo "${2%%XXX*}.${HOSTNAME}.${UID}.$$"
    }
fi
#生成临时文件
#${2}为该文件的后缀
tmpfile(){
    tmp=$(mktemp -u "${TMPDIR}/ffconf.XXXXXXXX")$2 &&
        (set -C; exec > $tmp) 2>/dev/null ||
        die "Unable to create temporary file in $TMPDIR."
    append TMPFILES $tmp
    eval $1=$tmp
}

trap 'rm -f -- $TMPFILES' EXIT
#各种临时文件
tmpfile TMPASM .asm
tmpfile TMPC   .c
tmpfile TMPCPP .cpp
tmpfile TMPE   $EXESUF
tmpfile TMPH   .h
tmpfile TMPO   .o
tmpfile TMPS   .S
tmpfile TMPSH  .sh
tmpfile TMPV   .ver

unset -f mktemp

chmod +x $TMPE

# make sure we can execute files in $TMPDIR
cat > $TMPSH 2>> $logfile <<EOF
#! /bin/sh
```

```
EOF
chmod +x $TMPSH >> $logfile 2>&1
if ! $TMPSH >> $logfile 2>&1; then
    cat <<EOF
Unable to create and execute files in $TMPDIR.  Set the TMPDIR environment
variable to another directory and make sure that it is not mounted noexec.
EOF
    die "Sanity test failed."
fi

filter_asflags=echo
filter_cflags=echo
filter_cppflags=echo
#检查编译器
if   $cc -v 2>&1 | grep -q '^gcc.*LLVM'; then
    cc_type=llvm_gcc
    cc_version=__VERSION__
    gcc_extra_ver=$(expr "$($cc --version | head -n1)" : '.*\((.*)\)')
    cc_ident="llvm-gcc $($cc -dumpversion) $gcc_extra_ver"
    CC_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
    AS_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
    speed_cflags='-O3'
    size_cflags='-Os'
elif $cc -v 2>&1 | grep -qi ^gcc; then
    cc_type=gcc
    cc_version=__VERSION__
    gcc_version=$($cc --version | head -n1)
    gcc_basever=$($cc -dumpversion)
    gcc_pkg_ver=$(expr "$gcc_version" : '[^ ]* \(([^)]*)\)')
    gcc_ext_ver=$(expr "$gcc_version" : ".*$gcc_pkg_ver $gcc_basever \\(.*\\)")
    cc_ident=$(cleanws "gcc $gcc_basever $gcc_pkg_ver $gcc_ext_ver")
    if ! $cc -dumpversion | grep -q '^2\.'; then
        CC_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
        AS_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
    fi
    speed_cflags='-O3'
    size_cflags='-Os'
elif $cc --version 2>/dev/null | grep -q Intel; then
    cc_type=icc
    cc_version="AV_STRINGIFY(__INTEL_COMPILER)"
    cc_ident=$($cc --version | head -n1)
    icc_version=$($cc -dumpversion)
    CC_DEPFLAGS='-MMD'
    AS_DEPFLAGS='-MMD'
    speed_cflags='-O3'
    size_cflags='-Os'
    noopt_cflags='-O1'
elif $cc -v 2>&1 | grep -q xlc; then
    cc_type=xlc
    cc_version="AV_STRINGIFY(__IBMC__)"
    cc_ident=$($cc -qversion 2>/dev/null | head -n1)
    speed_cflags='-O5'
    size_cflags='-O5 -qcompact'
elif $cc -V 2>/dev/null | grep -q Compaq; then
    cc_type=ccc
    cc_version="AV_STRINGIFY(__DECC_VER)"
    cc_ident=$($cc -V | head -n1 | cut -d' ' -f1-3)
    DEPFLAGS='$(CPPFLAGS) $(CFLAGS) -M'
    debuglevel=3
    add_ldflags -Wl,-z,now # calls to libots crash without this
    speed_cflags='-fast'
    size_cflags='-O1'
elif $cc --vsn 2>/dev/null | grep -q "ARM C/C++ Compiler"; then
    test -d "$sysroot" || die "No valid sysroot specified."
    cc_type=armcc
    cc_version="AV_STRINGIFY(__ARMCC_VERSION)"
    cc_ident=$($cc --vsn | head -n1)
    armcc_conf="$PWD/armcc.conf"
    $cc --arm_linux_configure                \
        --arm_linux_config_file="$armcc_conf" \
        --configure_sysroot="$sysroot"       \
        --configure_cpp_headers="$sysinclude" >>$logfile 2>&1 ||
        die "Error creating armcc configuration file."
    $cc --vsn | grep -q RVCT && armcc_opt=rvct || armcc_opt=armcc
    cc="$cc --arm_linux_config_file=$armcc_conf --translate_gcc"
    as_default="${cross_prefix}gcc"
    CC_DEPFLAGS='-MMD'
    AS_DEPFLAGS='-MMD'
    speed_cflags='-O3'
    size_cflags='-Os'
    filter_asflags="filter_out -W${armcc_opt}*"
elif $cc -version 2>/dev/null | grep -q TMS470; then
    cc_type=tms470
    cc_version="AV_STRINGIFY(__TI_COMPILER_VERSION__)"
    cc_ident=$($cc -version | head -n1 | tr -s ' ')
    cc="$cc --gcc --abi=eabi -eo=.o -mc -me"
    CC_O='-fr=$(@D)'
    as_default="${cross_prefix}gcc"
    ld_default="${cross_prefix}gcc"
    TMPO=$(basename $TMPC .c).o
    append TMPFILES $TMPO
    add_cflags -D__arm__ -D__ARM_ARCH__=4
```

```
    add_cflags -D__gnuc_va_list=va_list -D__USER_LABEL_PREFIX__=
    CC_DEPFLAGS='-ppa -ppd=$(@:.o=.d)'
    AS_DEPFLAGS='-MMD'
    speed_cflags='-O3 -mf=5'
    size_cflags='-O3 -mf=2'
    filter_cflags=tms470_flags
    tms470_flags(){
        for flag; do
            case $flag in
                -march=*|-mcpu=*)
                    case "${flag#*=}" in
                        armv7-a|cortex-a*)      echo -mv=7a8 ;;
                        armv7-r|cortex-r*)      echo -mv=7r4 ;;
                        armv7-m|cortex-m*)      echo -mv=7m3 ;;
                        armv6*|arm11*)          echo -mv=6   ;;
                        armv5*e|arm[79]*e*|arm9[24]6*|arm96*|arm102[26])
                                                echo -mv=5e  ;;
                        armv4*|arm7*|arm9[24]*) echo -mv=4   ;;
                    esac
                    ;;
                -mfpu=neon)     echo --float_support=vfpv3 --neon ;;
                -mfpu=vfp)      echo --float_support=vfpv2        ;;
                -mfpu=vfpv3)    echo --float_support=vfpv3        ;;
                -msoft-float)   echo --float_support=vfplib       ;;
                -O[0-3]|-mf=*)  echo $flag                        ;;
                -g)             echo -g -mn                       ;;
                -pds=*)         echo $flag                        ;;
            esac
        done
    }
elif $cc -v 2>&1 | grep -q clang; then
    cc_type=clang
    $cc -dM -E $TMPC | grep -q __clang_version__ &&
        cc_version=__clang_version__ || cc_version=__VERSION__
    cc_ident=$($cc --version | head -n1)
    CC_DEPFLAGS='-MMD'
    AS_DEPFLAGS='-MMD'
    speed_cflags='-O3'
    size_cflags='-Os'
elif $cc -V 2>&1 | grep -q Sun; then
    cc_type=suncc
    cc_version="AV_STRINGIFY(__SUNPRO_C)"
    cc_ident=$($cc -V 2>&1 | head -n1 | cut -d' ' -f 2-)
    DEPEND_CMD='$(DEPCC) $(DEPFLAGS) $< | sed -e "1s,^.*: ,$@: ," -e "\$$!s,\$$, \\\," -e "1!s,^.*: , ," > $(@:.o=.d)'
    DEPFLAGS='$(CPPFLAGS) $(CFLAGS) -xM1'
    add_ldflags -xc99
    speed_cflags='-O5'
    size_cflags='-O5 -xspace'
    filter_cflags=suncc_flags
    suncc_flags(){
        for flag; do
            case $flag in
                -march=*|-mcpu=*)
                    case "${flag#*=}" in
                        native)                 echo -xtarget=native       ;;
                        v9|niagara)             echo -xarch=sparc          ;;
                        ultrasparc)             echo -xarch=sparcvis       ;;
                        ultrasparc3|niagara2)   echo -xarch=sparcvis2      ;;
                        i586|pentium)           echo -xchip=pentium        ;;
                        i686|pentiumpro|pentium2) echo -xtarget=pentium_pro  ;;
                        pentium3*|c3-2)         echo -xtarget=pentium3     ;;
                        pentium-m)          echo -xarch=sse2 -xchip=pentium3 ;;
                        pentium4*)          echo -xtarget=pentium4          ;;
                        prescott|nocona)    echo -xarch=sse3 -xchip=pentium4 ;;
                        *-sse3)             echo -xarch=sse3                ;;
                        core2)              echo -xarch=ssse3 -xchip=core2   ;;
                        amdfam10|barcelona)       echo -xarch=sse4_1        ;;
                        athlon-4|athlon-[mx]p)    echo -xarch=ssea          ;;
                        k8|opteron|athlon64|athlon-fx)
                                                echo -xarch=sse2a          ;;
                        athlon*)                echo -xarch=pentium_proa   ;;
                    esac
                    ;;
                -std=c99)             echo -xc99              ;;
                -fomit-frame-pointer) echo -xregs=frameptr    ;;
                -fPIC)                echo -KPIC -xcode=pic32 ;;
                -W*,*)                echo $flag              ;;
                -f*-*|-W*)                                    ;;
                *)                    echo $flag              ;;
            esac
        done
    }
elif $cc -v 2>&1 | grep -q 'PathScale\|Path64'; then
    cc_type=pathscale
    cc_version=__PATHSCALE__
    cc_ident=$($cc -v 2>&1 | head -n1 | tr -d :)
    CC_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
    AS_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
    speed_cflags='-O2'
    size_cflags='-Os'
    filter_cflags='filter_out -Wdisabled-optimization'
elif $cc -v 2>&1 | grep -q Open64; then
```

```
elif $cc -v 2>&1 | grep -q OpenC64; then
    cc_type=open64
    cc_version=__OPEN64__
    cc_ident=$($cc -v 2>&1 | head -n1 | tr -d :)
    CC_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
    AS_DEPFLAGS='-MMD -MF $(@:.o=.d) -MT $@'
    speed_cflags='-O2'
    size_cflags='-Os'
    filter_cflags='filter_out -Wdisabled-optimization|-Wtype-limits|-fno-signed-zeros'
fi

test -n "$cc_type" && enable $cc_type ||
    warn "Unknown C compiler $cc, unable to select optimal CFLAGS"

: ${as_default:=$cc}
: ${dep_cc_default:=$cc}
: ${ld_default:=$cc}
set_default ar as dep_cc ld

test -n "$CC_DEPFLAGS" || CCDEP=$DEPEND_CMD
test -n "$CXX_DEPFLAGS" || CXXDEP=$DEPEND_CMD
test -n "$AS_DEPFLAGS" || ASDEP=$DEPEND_CMD

add_cflags $extra_cflags
add_cxxflags $extra_cxxflags
add_asflags $extra_cflags

if test -n "$sysroot"; then
    case "$cc_type" in
        gcc|llvm_gcc|clang)
            add_cppflags --sysroot="$sysroot"
            add_ldflags --sysroot="$sysroot"
        ;;
        tms470)
            add_cppflags -I"$sysinclude"
            add_ldflags  --sysroot="$sysroot"
        ;;
    esac
fi

if test "$cpu" = host; then
    enabled cross_compile && die "--cpu=host makes no sense when cross-compiling."

    case "$cc_type" in
        gcc|llvm_gcc)
            check_native(){
                $cc $1=native -v -c -o $TMPO $TMPC >$TMPE 2>&1 || return
                sed -n "/cc1.*$1=/{
                            s/.*$1=\\([^ ]*\\).*/\\1/
                            p
                            q
                        }" $TMPE
            }
            cpu=$(check_native -march || check_native -mcpu)
        ;;
    esac

    test "${cpu:-host}" = host && die "--cpu=host not supported with compiler $cc"
fi

# Deal with common $arch aliases
case "$arch" in
    arm*|iPad*)
        arch="arm"
    ;;
    mips|mipsel|IP*)
        arch="mips"
    ;;
    mips64*)
        arch="mips"
        subarch="mips64"
    ;;
    parisc|hppa)
        arch="parisc"
    ;;
    parisc64|hppa64)
        arch="parisc"
        subarch="parisc64"
    ;;
    "Power Macintosh"|ppc|powerpc|ppc64|powerpc64)
        arch="ppc"
    ;;
    s390|s390x)
        arch="s390"
    ;;
    sh4|sh)
        arch="sh4"
    ;;
    sun4u|sparc64)
        arch="sparc"
        subarch="sparc64"
    ;;
```

```
    ;;
    i[3-6]86|i86pc|BePC|x86pc|x86_64|x86_32|amd64)
        arch="x86"
    ;;
esac

is_in $arch $ARCH_LIST || warn "unknown architecture $arch"
enable $arch

# Add processor-specific flags
#根据CPU类型的不同，进行cpuflag的设置
if test "$cpu" = generic; then
    : do nothing
elif enabled ppc; then

    case $(tolower $cpu) in
        601|ppc601|powerpc601)
            cpuflags="-mcpu=601"
            disable altivec
        ;;
        603*|ppc603*|powerpc603*)
            cpuflags="-mcpu=603"
            disable altivec
        ;;
        604*|ppc604*|powerpc604*)
            cpuflags="-mcpu=604"
            disable altivec
        ;;
        g3|75*|ppc75*|powerpc75*)
            cpuflags="-mcpu=750 -mpowerpc-gfxopt"
            disable altivec
        ;;
        g4|745*|ppc745*|powerpc745*)
            cpuflags="-mcpu=7450 -mpowerpc-gfxopt"
        ;;
        74*|ppc74*|powerpc74*)
            cpuflags="-mcpu=7400 -mpowerpc-gfxopt"
        ;;
        g5|970|ppc970|powerpc970|power4*)
            cpuflags="-mcpu=970 -mpowerpc-gfxopt -mpowerpc64"
        ;;
        cell)
            cpuflags="-mcpu=cell"
            enable ldbrx
        ;;
        e500v2)
            cpuflags="-mcpu=8548 -mhard-float -mfloat-gprs=double"
            disable altivec
        ;;
        e500)
            cpuflags="-mcpu=8540 -mhard-float"
            disable altivec
        ;;
    esac
#X86架构
elif enabled x86; then

    case $cpu in
        i[345]86|pentium)
            cpuflags="-march=$cpu"
            disable mmx
        ;;
        # targets that do NOT support conditional mov (cmov)
        pentium-mmx|k6|k6-[23]|winchip-c6|winchip2|c3)
            cpuflags="-march=$cpu"
            disable cmov
        ;;
        # targets that do support conditional mov (cmov)
        i686|pentiumpro|pentium[23]|pentium-m|athlon|athlon-tbird|athlon-4|athlon-[mx]p|athlon64*|k8*|opteron*|athlon-fx|core2|amdfa
m10|barcelona|atom)
            cpuflags="-march=$cpu"
            enable cmov
            enable fast_cmov
        ;;
        # targets that do support conditional mov but on which it's slow
        pentium4|pentium4m|prescott|nocona)
            cpuflags="-march=$cpu"
            enable cmov
            disable fast_cmov
        ;;
    esac

elif enabled sparc; then

    case $cpu in
        niagara)
            cpuflags="-mcpu=$cpu"
            disable vis
        ;;
        sparc64)
            cpuflags="-mcpu=v9"
        ;;
```

```
    esac
#ARM架构
elif enabled arm; then

    case $cpu in
        armv*)
            cpuflags="-march=$cpu"
            subarch=$(echo $cpu | sed 's/[^a-z0-9]//g')
        ;;
        *)
            cpuflags="-mcpu=$cpu"
            case $cpu in
                cortex-a*)                              subarch=armv7a  ;;
                cortex-r*)                              subarch=armv7r  ;;
                cortex-m*)                enable thumb; subarch=armv7m  ;;
                arm11*)                                 subarch=armv6   ;;
                arm[79]*e*|arm9[24]6*|arm96*|arm102[26]) subarch=armv5te ;;
                armv4*|arm7*|arm9[24]*)                 subarch=armv4   ;;
            esac
        ;;
    esac

elif enabled alpha; then

    enabled ccc && cpuflags="-arch $cpu" || cpuflags="-mcpu=$cpu"

elif enabled bfin; then

    cpuflags="-mcpu=$cpu"

elif enabled mips; then

    cpuflags="-march=$cpu"

elif enabled avr32; then

    case $cpu in
        ap7[02]0[0-2])
            subarch="avr32_ap"
            cpuflags="-mpart=$cpu"
        ;;
        ap)
            subarch="avr32_ap"
            cpuflags="-march=$cpu"
        ;;
        uc3[ab]*)
            subarch="avr32_uc"
            cpuflags="-mcpu=$cpu"
        ;;
        uc)
            subarch="avr32_uc"
            cpuflags="-march=$cpu"
        ;;
    esac

fi

add_cflags $cpuflags
add_asflags $cpuflags

# compiler sanity check
# 用个简单的main()检查能不能用
check_exec <<EOF
int main(void){ return 0; }
EOF
if test "$?" != 0; then
    echo "$cc is unable to create an executable file."
    if test -z "$cross_prefix" && ! enabled cross_compile ; then
        echo "If $cc is a cross-compiler, use the --enable-cross-compile option."
        echo "Only do this if you know what cross compiling means."
    fi
    die "C compiler test failed."
fi

add_cppflags -D_ISOC99_SOURCE
add_cxxflags -D__STDC_CONSTANT_MACROS
check_cflags -std=c99
check_cc -D_FILE_OFFSET_BITS=64 <<EOF && add_cppflags -D_FILE_OFFSET_BITS=64
#include <stdlib.h>
EOF
check_cc -D_LARGEFILE_SOURCE <<EOF && add_cppflags -D_LARGEFILE_SOURCE
#include <stdlib.h>
EOF

check_host_cflags -std=c99
check_host_cflags -Wall
#32位系统指针变量占用32bit（4Byte）数据（32位寻址）
#64位系统指针变量占用64bit（4Byte）数据（64位寻址）
case "$arch" in
    alpha|ia64|mips|parisc|sparc)
        spic=$shared
```

```
    ;;
    x86)
        subarch="x86_32"
        check_cc <<EOF && subarch="x86_64"
        int test[(int)sizeof(char*) - 7];
EOF
        if test "$subarch" = "x86_64"; then
            spic=$shared
        fi
    ;;
    ppc)
        check_cc <<EOF && subarch="ppc64"
        int test[(int)sizeof(char*) - 7];
EOF
    ;;
esac

enable $subarch
enabled spic && enable pic

# 不同的操作系统 OS specific
# target-os参数
case $target_os in
    haiku)
        prefix_default="/boot/common"
        network_extralibs="-lnetwork"
        host_libs=
        ;;
    sunos)
        FFSERVERLDFLAGS=""
        SHFLAGS='-shared -Wl,-h,$$(@F)'
        enabled x86 && SHFLAGS="-mimpure-text $SHFLAGS"
        network_extralibs="-lsocket -lnsl"
        add_cppflags -D__EXTENSIONS__
        # When using suncc to build, the Solaris linker will mark
        # an executable with each instruction set encountered by
        # the Solaris assembler.  As our libraries contain their own
        # guards for processor-specific code, instead suppress
        # generation of the HWCAPS ELF section on Solaris x86 only.
        enabled_all suncc x86 && echo "hwcap_1 = OVERRIDE;" > mapfile && add_ldflags -Wl,-M,mapfile
        nm_opts='-P -g'
        ;;
    netbsd)
        disable symver
        oss_indev_extralibs="-lossaudio"
        oss_outdev_extralibs="-lossaudio"
        ;;
    openbsd)
        enable malloc_aligned
        # On OpenBSD 4.5. the compiler does not use PIC unless
        # explicitly using -fPIC. FFmpeg builds fine without PIC,
        # however the generated executable will not do anything
        # (simply quits with exit-code 1, no crash, no output).
        # Thus explicitly enable PIC here.
        enable pic
        disable symver
        SHFLAGS='-shared'
        oss_indev_extralibs="-lossaudio"
        oss_outdev_extralibs="-lossaudio"
        ;;
    dragonfly)
        enable malloc_aligned
        disable symver
        ;;
    freebsd)
        enable malloc_aligned
        ;;
    bsd/os)
        add_extralibs -lpoll -lgnugetopt
        strip="strip -d"
        ;;
#苹果Mac操作系统
    darwin)
        enable malloc_aligned
        #以前见过这个
        gas="gas-preprocessor.pl $cc"
        enabled ppc && add_asflags -force_cpusubtype_ALL
        SHFLAGS='-dynamiclib -Wl,-single_module -Wl,-install_name,$(SHLIBDIR)/$(SLIBNAME),-current_version,$(LIBVERSION),-compatibil
ity_version,$(LIBMAJOR)'
        enabled x86_32 && append SHFLAGS -Wl,-read_only_relocs,suppress
        strip="${strip} -x"
        add_ldflags -Wl,-dynamic,-search_paths_first
                #Mac下的动态库
        SLIBSUF=".dylib"
        SLIBNAME_WITH_VERSION='$(SLIBPREF)$(FULLNAME).$(LIBVERSION)$(SLIBSUF)'
        SLIBNAME_WITH_MAJOR='$(SLIBPREF)$(FULLNAME).$(LIBMAJOR)$(SLIBSUF)'
        FFSERVERLDFLAGS=-Wl,-bind_at_load
        #macho目标文件格式
        objformat="macho"
        enabled x86_64 && objformat="macho64"
        enabled_any pic shared ||
```

```
            { check_cflags -mdynamic-no-pic && add_asflags -mdynamic-no-pic; }
        ;;
#MinGW
    mingw32*)
        if test $target_os = "mingw32ce"; then
            disable network
        else
            target_os=mingw32
        fi
        LIBTARGET=i386
        if enabled x86_64; then
            enable malloc_aligned
            LIBTARGET=x64
        elif enabled arm; then
            LIBTARGET=arm-wince
        fi
        shlibdir_default="$bindir_default"
        SLIBPREF=""
                # Windows下动态库后缀
        SLIBSUF=".dll"
        SLIBNAME_WITH_VERSION='$(SLIBPREF)$(FULLNAME)-$(LIBVERSION)$(SLIBSUF)'
        SLIBNAME_WITH_MAJOR='$(SLIBPREF)$(FULLNAME)-$(LIBMAJOR)$(SLIBSUF)'
                # 借助lib.exe生成导出库lib
        SLIB_EXTRA_CMD='-lib.exe /machine:$(LIBTARGET) /def:$$(@:$(SLIBSUF)=.def) /out:$(SUBDIR)$(SLIBNAME:$(SLIBSUF)=.lib)'
        SLIB_INSTALL_NAME='$(SLIBNAME_WITH_MAJOR)'
        SLIB_INSTALL_LINKS=
        #额外的lib导入库
        SLIB_INSTALL_EXTRA_SHLIB='$(SLIBNAME:$(SLIBSUF)=.lib)'
        #额外的
        SLIB_INSTALL_EXTRA_LIB='lib$(SLIBNAME:$(SLIBSUF)=.dll.a) $(SLIBNAME_WITH_MAJOR:$(SLIBSUF)=.def)'
        SHFLAGS='-shared -Wl,--output-def,$$(@:$(SLIBSUF)=.def) -Wl,--out-implib,$(SUBDIR)lib$(SLIBNAME:$(SLIBSUF)=.dll.a) -Wl,--ena
ble-runtime-pseudo-reloc -Wl,--enable-auto-image-base'
        # windows PE格式
        objformat="win32"
        enable dos_paths
        check_cflags -fno-common
        check_cpp_condition _mingw.h "defined (__MINGW64_VERSION_MAJOR) || (__MINGW32_MAJOR_VERSION > 3) \
                                      || (__MINGW32_MAJOR_VERSION == 3 && __MINGW32_MINOR_VERSION >= 15)" ||
                die "ERROR: MinGW runtime version must be >= 3.15."
        add_cppflags -U__STRICT_ANSI__
        ;;
#Cygwin
    cygwin*)
        target_os=cygwin
        shlibdir_default="$bindir_default"
        SLIBPREF="cyg"
        SLIBSUF=".dll"
        SLIBNAME_WITH_VERSION='$(SLIBPREF)$(FULLNAME)-$(LIBVERSION)$(SLIBSUF)'
        SLIBNAME_WITH_MAJOR='$(SLIBPREF)$(FULLNAME)-$(LIBMAJOR)$(SLIBSUF)'
        SHFLAGS='-shared -Wl,--enable-auto-image-base'
        objformat="win32"
        enable dos_paths
        check_cflags -fno-common
        add_cppflags -U__STRICT_ANSI__
        ;;
    *-dos|freedos|opendos)
        network_extralibs="-lsocket"
        objformat="coff"
        enable dos_paths
        add_cppflags -U__STRICT_ANSI__
        ;;
#Linux操作系统
    linux)
        add_cppflags -D_POSIX_C_SOURCE=200112 -D_XOPEN_SOURCE=600
        enable dv1394
        ;;
    irix*)
        target_os=irix
        ranlib="echo ignoring ranlib"
        ;;
    os/2*)
        strip="lxlite -CS"
        ln_s="cp -f"
        objformat="aout"
        add_cppflags -D_GNU_SOURCE
        add_ldflags -Zomf -Zbin-files -Zargs-wild -Zmap
        SHFLAGS='$(SUBDIR)$(NAME).def -Zdll -Zomf'
        FFSERVERLDFLAGS=""
        LIBSUF="_s.a"
        SLIBPREF=""
        SLIBSUF=".dll"
        SLIBNAME_WITH_VERSION='$(SLIBPREF)$(NAME)-$(LIBVERSION)$(SLIBSUF)'
        SLIBNAME_WITH_MAJOR='$(SLIBPREF)$(shell echo $(NAME) | cut -c1-6)$(LIBMAJOR)$(SLIBSUF)'
        SLIB_CREATE_DEF_CMD='echo LIBRARY $(SLIBNAME_WITH_MAJOR) INITINSTANCE TERMINSTANCE > $(SUBDIR)$(NAME).def; \
          echo PROTMODE >> $(SUBDIR)$(NAME).def; \
          echo CODE PRELOAD MOVEABLE DISCARDABLE >> $(SUBDIR)$(NAME).def; \
          echo DATA PRELOAD MOVEABLE MULTIPLE NONSHARED >> $(SUBDIR)$(NAME).def; \
          echo EXPORTS >> $(SUBDIR)$(NAME).def; \
          emxexp -o $(OBJS) >> $(SUBDIR)$(NAME).def'
        SLIB_EXTRA_CMD='emximp -o $(SUBDIR)$(LIBPREF)$(NAME)_dll.a $(SUBDIR)$(NAME).def; \
          emximp -o $(SUBDIR)$(LIBPREF)$(NAME)_dll.lib $(SUBDIR)$(NAME).def;'
        SLIB_INSTALL_EXTRA_LIB='$(LIBPREF)$(NAME)_dll.a $(LIBPREF)$(NAME)_dll.lib'
```

```
        SLIB_INSTALL_EXTRA_LIB='$(LIBPREF)$(NAME)_dll.a $(LIBPREF)$(NAME)_dll.lib'
        enable dos_paths
        enable_weak os2threads
        ;;
    gnu/kfreebsd)
        add_cppflags -D_POSIX_C_SOURCE=200112 -D_XOPEN_SOURCE=600 -D_BSD_SOURCE
        ;;
    gnu)
        add_cppflags -D_POSIX_C_SOURCE=200112 -D_XOPEN_SOURCE=600
        ;;
    qnx)
        add_cppflags -D_QNX_SOURCE
        network_extralibs="-lsocket"
        ;;
    symbian)
        SLIBSUF=".dll"
        enable dos_paths
        add_cflags --include=$sysinclude/gcce/gcce.h -fvisibility=default
        add_cppflags -D__GCCE__ -D__SYMBIAN32__ -DSYMBIAN_OE_POSIX_SIGNALS
        add_ldflags -Wl,--target1-abs,--no-undefined \
                    -Wl,-Ttext,0x80000,-Tdata,0x1000000 -shared \
                    -Wl,--entry=_E32Startup -Wl,-u,_E32Startup
        add_extralibs -l:eexe.lib -l:usrt2_2.lib -l:dfpaeabi.dso \
                      -l:drtaeabi.dso -l:scppnwdl.dso -lsupc++ -lgcc \
                      -l:libc.dso -l:libm.dso -l:euser.dso -l:libcrt0.lib
        ;;
    none)
        ;;
    *)
        die "Unknown OS '$target_os'."
        ;;
esac

echo "config:$arch:$subarch:$cpu:$target_os:$cc_ident:$FFMPEG_CONFIGURATION" >config.fate

check_cpp_condition stdlib.h "defined(__PIC__) || defined(__pic__) || defined(PIC)" && enable pic

set_default $PATHS_LIST

# we need to build at least one lib type
if ! enabled_any static shared; then
    cat <<EOF
At least one library type must be built.
Specify --enable-static to build the static libraries or --enable-shared to
build the shared libraries as well. To only build the shared libraries specify
--disable-static in addition to --enable-shared.
EOF
    exit 1;
fi
#不符合License则立刻结束
die_license_disabled() {
    enabled $1 || { enabled $2 && die "$2 is $1 and --enable-$1 is not specified."; }
}
#检查License
#GPL
die_license_disabled gpl libcdio
die_license_disabled gpl libx264
die_license_disabled gpl libxavs
die_license_disabled gpl libxvid
die_license_disabled gpl x11grab
#nonfree
die_license_disabled nonfree libaacplus
die_license_disabled nonfree libfaac
die_license_disabled nonfree openssl
#Version3
die_license_disabled version3 libopencore_amrnb
die_license_disabled version3 libopencore_amrwb
die_license_disabled version3 libvo_aacenc
die_license_disabled version3 libvo_amrwbenc

enabled version3 && { enabled gpl && enable gplv3 || enable lgplv3; }

disabled optimizations || check_cflags -fomit-frame-pointer
#添加fPIC
enable_pic() {
    enable pic
    add_cppflags -DPIC
    add_cflags   -fPIC
    add_asflags  -fPIC
}

enabled pic && enable_pic

check_cc <<EOF || die "Symbol mangling check failed."
int ff_extern;
EOF
sym=$($nm $nm_opts $TMPO | awk '/ff_extern/{ print substr($0, match($0, /[^ \t]*ff_extern/)) }')
extern_prefix=${sym%%ff_extern*}

check_cc <<EOF && enable inline_asm
void foo(void) { __asm__ volatile ("" ::); }
EOF
```

```
EOF

_restrict=
for restrict_keyword in restrict __restrict__ __restrict; do
    check_cc <<EOF && _restrict=$restrict_keyword && break
void foo(char * $restrict_keyword p);
EOF
done

check_cc <<EOF && enable attribute_packed
struct { int x; } __attribute__((packed)) x;
EOF

check_cc <<EOF && enable attribute_may_alias
union { int x; } __attribute__((may_alias)) x;
EOF

check_cc <<EOF || die "endian test failed"
unsigned int endian = 'B' << 24 | 'I' << 16 | 'G' << 8 | 'E';
EOF
od -t x1 $TMPO | grep -q '42 *49 *47 *45' && enable bigendian

if enabled alpha; then

    check_cflags -mieee

elif enabled arm; then

    enabled thumb && check_cflags -mthumb || check_cflags -marm
    nogas=die

    if     check_cpp_condition stddef.h "defined __ARM_PCS_VFP"; then
        enable vfp_args
    elif ! check_cpp_condition stddef.h "defined __ARM_PCS || defined __SOFTFP__"; then
        case "${cross_prefix:-$cc}" in
            *hardfloat*)         enable vfp_args;   fpabi=vfp ;;
            *) check_ld "cc" <<EOF && enable vfp_args && fpabi=vfp || fpabi=soft ;;
__asm__ (".eabi_attribute 28, 1");
int main(void) { return 0; }
EOF
        esac
        warn "Compiler does not indicate floating-point ABI, guessing $fpabi."
    fi

    enabled armv5te && check_asm armv5te '"qadd r0, r0, r0"'
    enabled armv6   && check_asm armv6   '"sadd16 r0, r0, r0"'
    enabled armv6t2 && check_asm armv6t2 '"movt r0, #0"'
    enabled armvfp  && check_asm armvfp  '"fadds s0, s0, s0"'
    enabled iwmmxt  && check_asm iwmmxt  '"wunpckelub wr6, wr4"'
    enabled neon    && check_asm neon    '"vadd.i16 q0, q0, q0"'
    enabled vfpv3   && check_asm vfpv3   '"vmov.f32 s0, #1.0"'

    check_asm asm_mod_y '"vmul.i32 d0, d0, %y0" :: "x"(0)'

    enabled_all armv6t2 shared !pic && enable_pic

elif enabled mips; then

    check_asm loongson '"dmult.g $1, $2, $3"'
    enabled mmi     && check_asm mmi     '"lq $2, 0($2)"'

elif enabled ppc; then

    enable local_aligned_8 local_aligned_16

    check_asm dcbzl     '"dcbzl 0, %0" :: "r"(0)'
    check_asm ibm_asm   '"add 0, 0, 0"'
    check_asm ppc4xx    '"maclhw r10, r11, r12"'
    check_asm xform_asm '"lwzx %1, %y0" :: "Z"(*(int*)0), "r"(0)'

    # AltiVec flags: The FSF version of GCC differs from the Apple version
    if enabled altivec; then
        nogas=warn
        check_cflags -maltivec -mabi=altivec &&
        { check_header altivec.h && inc_altivec_h="#include <altivec.h>" ; } ||
        check_cflags -faltivec

        # check if our compiler supports Motorola AltiVec C API
        check_cc <<EOF || disable altivec
$inc_altivec_h
int main(void) {
    vector signed int v1, v2, v3;
    v1 = vec_add(v2,v3);
    return 0;
}
EOF

        # check if our compiler supports braces for vector declarations
        check_cc <<EOF || die "You need a compiler that supports {} in AltiVec vector declarations."
$inc_altivec_h
int main (void) { (vector int) {1}; return 0; }
EOF
```

```
    fi

elif enabled sparc; then

    enabled vis && check_asm vis '"pdist %f0, %f0, %f0"' -mcpu=ultrasparc &&
        add_cflags -mcpu=ultrasparc -mtune=ultrasparc

elif enabled x86; then

    enable local_aligned_8 local_aligned_16

    # check whether EBP is available on x86
    # As 'i' is stored on the stack, this program will crash
    # if the base pointer is used to access it because the
    # base pointer is cleared in the inline assembly code.
    check_exec_crash <<EOF && enable ebp_available
    volatile int i=0;
    __asm__ volatile (
        "xorl %%ebp, %%ebp"
    ::: "%ebp");
    return i;
EOF

    # check whether EBX is available on x86
    check_asm ebx_available '"":"b"(0)' &&
        check_asm ebx_available '"":::"%ebx"'

    # check whether xmm clobbers are supported
    check_asm xmm_clobbers '"":::"%xmm0"'

    # check whether binutils is new enough to compile SSSE3/MMX2
    enabled ssse3 && check_asm ssse3 '"pabsw %xmm0, %xmm0"'
    enabled mmx2  && check_asm mmx2  '"pmaxub %mm0, %mm1"'

    if ! disabled_any asm mmx yasm; then
        if check_cmd $yasmexe --version; then
            enabled x86_64 && yasm_extra="-m amd64"
            yasm_debug="-g dwarf2"
        elif check_cmd nasm -v; then
            yasmexe=nasm
            yasm_debug="-g -F dwarf"
            enabled x86_64 && test "$objformat" = elf && objformat=elf64
        fi

        YASMFLAGS="-f $objformat $yasm_extra"
        enabled pic               && append YASMFLAGS "-DPIC"
        test -n "$extern_prefix"  && append YASMFLAGS "-DPREFIX"
        case "$objformat" in
            elf*) enabled debug && append YASMFLAGS $yasm_debug ;;
        esac

        check_yasm "pextrd [eax], xmm0, 1" && enable yasm ||
            die "yasm not found, use --disable-yasm for a crippled build"
        check_yasm "vextractf128 xmm0, ymm0, 0" || disable avx
    fi

    case "$cpu" in
        athlon*|opteron*|k8*|pentium|pentium-mmx|prescott|nocona|atom|geode)
            disable fast_clz
        ;;
    esac

fi

if enabled asm; then
    as=${gas:=$as}
    check_asm gnu_as '".macro m n\n\\n:.int 0\n.endm\nm x"' ||
        $nogas "GNU assembler not found, install gas-preprocessor"
fi

check_ldflags -Wl,--as-needed

if check_func dlopen; then
    ldl=
elif check_func dlopen -ldl; then
    ldl=-ldl
fi
#网络socket
if enabled network; then
    check_type "sys/types.h sys/socket.h" socklen_t
    check_type netdb.h "struct addrinfo"
    check_type netinet/in.h "struct ipv6_mreq" -D_DARWIN_C_SOURCE
    check_type netinet/in.h "struct sockaddr_in6"
    check_type "sys/types.h sys/socket.h" "struct sockaddr_storage"
    check_struct "sys/types.h sys/socket.h" "struct sockaddr" sa_len
    # Prefer arpa/inet.h over winsock2
    if check_header arpa/inet.h ; then
        check_func closesocket
    elif check_header winsock2.h ; then
        check_func_headers winsock2.h closesocket -lws2 && \
            network_extralibs="-lws2" || \
```

```
        { check_func_headers winsock2.h closesocket -lws2_32 && \
            network_extralibs="-lws2_32"; }
        check_type ws2tcpip.h socklen_t
        check_type ws2tcpip.h "struct addrinfo"
        check_type ws2tcpip.h "struct ipv6_mreq"
        check_type ws2tcpip.h "struct sockaddr_in6"
        check_type ws2tcpip.h "struct sockaddr_storage"
        check_struct winsock2.h "struct sockaddr" sa_len
    else
        disable network
    fi
fi

# Solaris has nanosleep in -lrt, OpenSolaris no longer needs that
check_func nanosleep || { check_func nanosleep -lrt && add_extralibs -lrt; }
#检查函数
check_func  fcntl
check_func  fork
check_func  getaddrinfo $network_extralibs
check_func  gethrtime
check_func  getrusage
check_struct "sys/time.h sys/resource.h" "struct rusage" ru_maxrss
check_func  inet_aton $network_extralibs
check_func  isatty
check_func  localtime_r
check_func  ${malloc_prefix}memalign            && enable memalign
check_func  mkstemp
check_func  mmap
check_func  ${malloc_prefix}posix_memalign      && enable posix_memalign
check_func  setrlimit
check_func  strerror_r
check_func  strptime
check_func  sched_getaffinity
check_func  sysconf
check_func  sysctl
check_func_headers conio.h kbhit
check_func_headers windows.h PeekNamedPipe
check_func_headers io.h setmode
check_func_headers lzo/lzo1x.h lzo1x_999_compress
check_lib2 "windows.h psapi.h" GetProcessMemoryInfo -lpsapi
check_func_headers windows.h GetProcessAffinityMask
check_func_headers windows.h GetProcessTimes
check_func_headers windows.h MapViewOfFile
check_func_headers windows.h VirtualAlloc
#检查头文件
check_header dlfcn.h
check_header dxva2api.h -D_WIN32_WINNT=0x0600
check_header libcrystalhd/libcrystalhd_if.h
check_header malloc.h
check_header poll.h
check_header sys/mman.h
check_header sys/param.h
check_header sys/resource.h
check_header sys/select.h
check_header termios.h
check_header vdpau/vdpau.h
check_header vdpau/vdpau_x11.h
check_header X11/extensions/XvMClib.h
check_header asm/types.h

disabled  zlib || check_lib   zlib.h      zlibVersion -lz   || disable  zlib
disabled bzlib || check_lib2 bzlib.h BZ2_bzlibVersion -lbz2 || disable bzlib

# check for VDA header
if ! disabled vda; then
    if check_header VideoDecodeAcceleration/VDADecoder.h; then
        enable vda
        add_extralibs -framework CoreFoundation -framework VideoDecodeAcceleration -framework QuartzCore
    fi
fi

if ! disabled w32threads && ! enabled pthreads; then
    check_func _beginthreadex && enable w32threads
fi

# check for some common methods of building with pthread support
# do this before the optional library checks as some of them require pthreads
if ! disabled pthreads && ! enabled w32threads && ! enabled os2threads; then
    enable pthreads
    if check_func pthread_create; then
        :
    elif check_func pthread_create -pthread; then
        add_cflags -pthread
        add_extralibs -pthread
    elif check_func pthread_create -pthreads; then
        add_cflags -pthreads
        add_extralibs -pthreads
    elif check_func pthread_create -lpthreadGC2; then
        add_extralibs -lpthreadGC2
    elif ! check_lib pthread.h pthread_create -lpthread; then
        disable pthreads
```

```
    fi
fi

for thread in $THREADS_LIST; do
    if enabled $thread; then
        test -n "$thread_type" &&
            die "ERROR: Only one thread type must be selected." ||
            thread_type="$thread"
    fi
done

check_lib math.h sin -lm && LIBM="-lm"
disabled crystalhd || check_lib libcrystalhd/libcrystalhd_if.h DtsCrystalHDVersion -lcrystalhd || disable crystalhd
enabled vaapi && require vaapi va/va.h vaInitialize -lva
#检查数学函数
check_mathfunc cbrtf
check_mathfunc exp2
check_mathfunc exp2f
check_mathfunc llrint
check_mathfunc llrintf
check_mathfunc log2
check_mathfunc log2f
check_mathfunc lrint
check_mathfunc lrintf
check_mathfunc round
check_mathfunc roundf
check_mathfunc trunc
check_mathfunc truncf

#检查第三方类库
# these are off by default, so fail if requested and not available
#require()函数参数的规范：（名称，头文件，函数名，附加选项）
#require2()函数参数规范类似
enabled avisynth   && require2 vfw32 "windows.h vfw.h" AVIFileInit -lavifil32
enabled frei0r     && { check_header frei0r.h || die "ERROR: frei0r.h header not found"; }
enabled gnutls     && require_pkg_config gnutls gnutls/gnutls.h gnutls_global_init
enabled libaacplus && require  "libaacplus >= 2.0.0" aacplus.h aacplusEncOpen -laacplus
enabled libass     && require_pkg_config libass ass/ass.h ass_library_init
enabled libcelt    && require libcelt celt/celt.h celt_decode -lcelt0 &&
                      { check_lib celt/celt.h celt_decoder_create_custom -lcelt0 ||
                        die "ERROR: libcelt version must be >= 0.11.0."; }
enabled libdc1394  && require_pkg_config libdc1394-2 dc1394/dc1394.h dc1394_new
enabled libdirac   && require_pkg_config dirac                          \
    "libdirac_decoder/dirac_parser.h libdirac_encoder/dirac_encoder.h"  \
    "dirac_decoder_init dirac_encoder_init"
#测试libfaac
enabled libfaac    && require2 libfaac "stdint.h faac.h" faacEncGetVersion -lfaac
enabled libfreetype && require_pkg_config freetype2 "ft2build.h freetype/freetype.h" FT_Init_FreeType
enabled libgsm     && require  libgsm gsm/gsm.h gsm_create -lgsm
enabled libmodplug && require  libmodplug libmodplug/modplug.h ModPlug_Load -lmodplug
enabled libmp3lame && require  "libmp3lame >= 3.98.3" lame/lame.h lame_set_VBR_quality -lmp3lame
enabled libnut     && require  libnut libnut.h nut_demuxer_init -lnut
enabled libopencore_amrnb  && require libopencore_amrnb opencore-amrnb/interf_dec.h Decoder_Interface_init -lopencore-amrnb
enabled libopencore_amrwb  && require libopencore_amrwb opencore-amrwb/dec_if.h D_IF_init -lopencore-amrwb
enabled libopencv  && require_pkg_config opencv opencv/cxcore.h cvCreateImageHeader
enabled libopenjpeg && require libopenjpeg openjpeg.h opj_version -lopenjpeg
enabled libpulse && require_pkg_config libpulse-simple pulse/simple.h pa_simple_new
enabled librtmp    && require_pkg_config librtmp librtmp/rtmp.h RTMP_Socket
enabled libschroedinger && require_pkg_config schroedinger-1.0 schroedinger/schro.h schro_init
enabled libspeex   && require  libspeex speex/speex.h speex_decoder_init -lspeex
enabled libstagefright_h264  && require_cpp libstagefright_h264 "binder/ProcessState.h media/stagefright/MetaData.h
    media/stagefright/MediaBufferGroup.h media/stagefright/MediaDebug.h media/stagefright/MediaDefs.h
    media/stagefright/OMXClient.h media/stagefright/OMXCodec.h" android::OMXClient -lstagefright -lmedia -lutils -lbinder
enabled libtheora  && require  libtheora theora/theoraenc.h th_info_init -ltheoraenc -ltheoradec -logg
enabled libutvideo    && require_cpp utvideo "stdint.h stdlib.h utvideo/utvideo.h utvideo/Codec.h" 'CCodec*' -lutvideo -lstdc++
enabled libv4l2    && require_pkg_config libv4l2 libv4l2.h v4l2_ioctl
enabled libvo_aacenc && require libvo_aacenc vo-aacenc/voAAC.h voGetAACEncAPI -lvo-aacenc
enabled libvo_amrwbenc && require libvo_amrwbenc vo-amrwbenc/enc_if.h E_IF_init -lvo-amrwbenc
enabled libvorbis  && require  libvorbis vorbis/vorbisenc.h vorbis_info_init -lvorbisenc -lvorbis -logg
enabled libvpx     && {
    enabled libvpx_decoder && { check_lib2 "vpx/vpx_decoder.h vpx/vp8dx.h" vpx_codec_dec_init_ver -lvpx ||
                                die "ERROR: libvpx decoder version must be >=0.9.1"; }
    enabled libvpx_encoder && { check_lib2 "vpx/vpx_encoder.h vpx/vp8cx.h" "vpx_codec_enc_init_ver VPX_CQ" -lvpx ||
                                die "ERROR: libvpx encoder version must be >=0.9.6"; } }
#测试libx264
enabled libx264    && require  libx264 x264.h x264_encoder_encode -lx264 &&
                      { check_cpp_condition x264.h "X264_BUILD >= 118" ||
                        die "ERROR: libx264 version must be >= 0.118."; }
enabled libxavs    && require  libxavs xavs.h xavs_encoder_encode -lxavs
enabled libxvid    && require  libxvid xvid.h xvid_global -lxvidcore
enabled openal     && { { for al_libs in "${OPENAL_LIBS}" "-lopenal" "-lOpenAL32"; do
                          check_lib 'AL/al.h' alGetError "${al_libs}" && break; done } ||
                          die "ERROR: openal not found"; } &&
                      { check_cpp_condition "AL/al.h" "defined(AL_VERSION_1_1)" ||
                        die "ERROR: openal version must be 1.1 or compatible"; }
enabled mlib       && require  mediaLib mlib_types.h mlib_VectorSub_S16_U8_Mod -lmlib
enabled openssl    && { check_lib openssl/ssl.h SSL_library_init -lssl -lcrypto ||
                          check_lib openssl/ssl.h SSL_library_init -lssl32 -leay32 ||
                          check_lib openssl/ssl.h SSL_library_init -lssl -lcrypto -lws2_32 -lgdi32 ||
                          die "ERROR: openssl not found"; }
#检查SDL
```

```
SDL_CONFIG="${cross_prefix}sdl-config"
if check_pkg_config sdl SDL_version.h SDL_Linked_Version; then
    check_cpp_condition SDL.h "(SDL_MAJOR_VERSION<<16 | SDL_MINOR_VERSION<<8 | SDL_PATCHLEVEL) >= 0x010201" $sdl_cflags &&
    enable sdl &&
    check_struct SDL.h SDL_VideoInfo current_w $sdl_cflags && enable sdl_video_size
else
  if "${SDL_CONFIG}" --version > /dev/null 2>&1; then
    sdl_cflags=$("${SDL_CONFIG}" --cflags)
    sdl_libs=$("${SDL_CONFIG}" --libs)
    check_func_headers SDL_version.h SDL_Linked_Version $sdl_cflags $sdl_libs &&
    check_cpp_condition SDL.h "(SDL_MAJOR_VERSION<<16 | SDL_MINOR_VERSION<<8 | SDL_PATCHLEVEL) >= 0x010201" $sdl_cflags &&
    enable sdl &&
    check_struct SDL.h SDL_VideoInfo current_w $sdl_cflags && enable sdl_video_size
  fi
fi
enabled sdl && add_cflags $sdl_cflags && add_extralibs $sdl_libs

texi2html -version > /dev/null 2>&1 && enable texi2html || disable texi2html
makeinfo --version > /dev/null 2>&1 && enable makeinfo  || disable makeinfo
#检查头文件
check_header linux/fb.h
check_header linux/videodev.h
check_header linux/videodev2.h
check_struct linux/videodev2.h "struct v4l2_frmivalenum" discrete

check_header sys/videoio.h

check_func_headers "windows.h vfw.h" capCreateCaptureWindow "$vfwcap_indev_extralibs"
# check that WM_CAP_DRIVER_CONNECT is defined to the proper value
# w32api 3.12 had it defined wrong
check_cpp_condition vfw.h "WM_CAP_DRIVER_CONNECT > WM_USER" && enable vfwcap_defines

check_type "dshow.h" IBaseFilter

# check for ioctl_meteor.h, ioctl_bt848.h and alternatives
{ check_header dev/bktr/ioctl_meteor.h &&
  check_header dev/bktr/ioctl_bt848.h; } ||
{ check_header machine/ioctl_meteor.h &&
  check_header machine/ioctl_bt848.h; } ||
{ check_header dev/video/meteor/ioctl_meteor.h &&
  check_header dev/video/bktr/ioctl_bt848.h; } ||
check_header dev/ic/bt8xx.h

check_header sndio.h
if check_struct sys/soundcard.h audio_buf_info bytes; then
    enable_safe sys/soundcard.h
else
    check_cc -D__BSD_VISIBLE -D__XSI_VISIBLE <<EOF && add_cppflags -D__BSD_VISIBLE -D__XSI_VISIBLE && enable_safe sys/soundcard.h
    #include <sys/soundcard.h>
    audio_buf_info abc;
EOF
fi
check_header soundcard.h

enabled_any alsa_indev alsa_outdev && check_lib2 alsa/asoundlib.h snd_pcm_htimestamp -lasound

enabled jack_indev && check_lib2 jack/jack.h jack_client_open -ljack && check_func sem_timedwait

enabled_any sndio_indev sndio_outdev && check_lib2 sndio.h sio_open -lsndio

enabled libcdio &&
    check_lib2 "cdio/cdda.h cdio/paranoia.h" cdio_cddap_open "-lcdio_paranoia -lcdio_cdda -lcdio"

enabled x11grab                         &&
check_header X11/Xlib.h                 &&
check_header X11/extensions/XShm.h      &&
check_header X11/extensions/Xfixes.h    &&
check_func XOpenDisplay -lX11           &&
check_func XShmCreateImage -lX11 -lXext &&
check_func XFixesGetCursorImage -lX11 -lXext -lXfixes

if ! disabled vaapi; then
    check_lib va/va.h vaInitialize -lva && {
        check_cpp_condition va/va_version.h "VA_CHECK_VERSION(0,32,0)" ||
        warn "Please upgrade to VA-API >= 0.32 if you would like full VA-API support.";
    } || disable vaapi
fi

if ! disabled vdpau && enabled vdpau_vdpau_h; then
check_cpp_condition \
    vdpau/vdpau.h "defined VDP_DECODER_PROFILE_MPEG4_PART2_ASP" ||
    { echolog "Please upgrade to libvdpau >= 0.2 if you would like vdpau support." &&
      disable vdpau; }
fi

enabled debug && add_cflags -g"$debuglevel" && add_asflags -g"$debuglevel"
enabled coverage && add_cflags "-fprofile-arcs -ftest-coverage" && add_ldflags "-fprofile-arcs -ftest-coverage"
test -n "$valgrind" && target_exec="$valgrind --error-exitcode=1 --malloc-fill=0x2a --track-origins=yes --leak-check=full --gen-suppressions=all --suppressions=$source_path/tests/fate-valgrind.supp"
#添加一些编译选项
# add some useful compiler flags if supported
check_cflags -Wdeclaration-after-statement
```

```
check_cflags -Wdeclaration-after-statement
check_cflags -Wall
check_cflags -Wno-parentheses
check_cflags -Wno-switch
check_cflags -Wno-format-zero-length
check_cflags -Wdisabled-optimization
check_cflags -Wpointer-arith
check_cflags -Wredundant-decls
check_cflags -Wno-pointer-sign
check_cflags -Wcast-qual
check_cflags -Wwrite-strings
check_cflags -Wtype-limits
check_cflags -Wundef
check_cflags -Wmissing-prototypes
check_cflags -Wno-pointer-to-int-cast
check_cflags -Wstrict-prototypes
enabled extra_warnings && check_cflags -Winline

# add some linker flags
check_ldflags -Wl,--warn-common
check_ldflags -Wl,-rpath-link=libpostproc:libswresample:libswscale:libavfilter:libavdevice:libavformat:libavcodec:libavutil
test_ldflags -Wl,-Bsymbolic && append SHFLAGS -Wl,-Bsymbolic

echo "X{};" > $TMPV
if test_ldflags -Wl,--version-script,$TMPV; then
    append SHFLAGS '-Wl,--version-script,\$(SUBDIR)lib\$(NAME).ver'
    check_cc <<EOF && enable symver_asm_label
void ff_foo(void) __asm__ ("av_foo@VERSION");
void ff_foo(void) { ${inline_asm+__asm__($quotes);} }
EOF
    check_cc <<EOF && enable symver_gnu_asm
__asm__(".symver ff_foo,av_foo@VERSION");
void ff_foo(void) {}
EOF
fi

if [ -n "$optflags" ]; then
    add_cflags $optflags
elif enabled small; then
    add_cflags $size_cflags
elif enabled optimizations; then
    add_cflags $speed_cflags
else
    add_cflags $noopt_cflags
fi
check_cflags -fno-math-errno
check_cflags -fno-signed-zeros
check_cc -mno-red-zone <<EOF && noredzone_flags="-mno-red-zone"
int x;
EOF


if enabled icc; then
    # Just warnings, no remarks
    check_cflags -w1
    # -wd: Disable following warnings
    # 144, 167, 556: -Wno-pointer-sign
    # 1292: attribute "foo" ignored
    # 10006: ignoring unknown option -fno-signed-zeros
    # 10148: ignoring unknown option -Wno-parentheses
    # 10156: ignoring option '-W'; no argument required
    check_cflags -wd144,167,556,1292,10006,10148,10156
    # 11030: Warning unknown option --as-needed
    # 10156: ignoring option '-export'; no argument required
    check_ldflags -wd10156,11030
    # Allow to compile with optimizations
    check_ldflags -march=$cpu
    # icc 11.0 and 11.1 work with ebp_available, but don't pass the test
    enable ebp_available
    if enabled x86_32; then
        test ${icc_version%%.*} -ge 11 && \
            check_cflags -falign-stack=maintain-16-byte || \
            disable aligned_stack
    fi
elif enabled ccc; then
    # disable some annoying warnings
    add_cflags -msg_disable cvtu32to64
    add_cflags -msg_disable embedcomment
    add_cflags -msg_disable needdconstext
    add_cflags -msg_disable nomainieee
    add_cflags -msg_disable ptrmismatch1
    add_cflags -msg_disable unreachcode
elif enabled gcc; then
    check_cflags -fno-tree-vectorize
    check_cflags -Werror=implicit-function-declaration
    check_cflags -Werror=missing-prototypes
elif enabled llvm_gcc; then
    check_cflags -mllvm -stack-alignment=16
elif enabled clang; then
    check_cflags -mllvm -stack-alignment=16
    check_cflags -Qunused-arguments
elif enabled armcc; then
```

```
elif enabled armcc; then
    # 2523: use of inline assembler is deprecated
    add_cflags -W${armcc_opt},--diag_suppress=2523
    add_cflags -W${armcc_opt},--diag_suppress=1207
    add_cflags -W${armcc_opt},--diag_suppress=1293 # assignment in condition
    add_cflags -W${armcc_opt},--diag_suppress=3343 # hardfp compat
    add_cflags -W${armcc_opt},--diag_suppress=167  # pointer sign
    add_cflags -W${armcc_opt},--diag_suppress=513  # pointer sign
elif enabled tms470; then
    add_cflags -pds=824 -pds=837
elif enabled pathscale; then
    add_cflags -fstrict-overflow -OPT:wrap_around_unsafe_opt=OFF
fi

enabled_any $THREADS_LIST      && enable threads

check_deps $CONFIG_LIST       \
           $CONFIG_EXTRA      \
           $HAVE_LIST         \
           $ALL_COMPONENTS    \
           $ALL_TESTS         \

enabled asm || { arch=c; disable $ARCH_LIST $ARCH_EXT_LIST; }

if test $target_os = "haiku"; then
    disable memalign
    disable posix_memalign
fi

! enabled_any memalign posix_memalign malloc_aligned &&
    enabled_any $need_memalign && enable memalign_hack

#在控制台输出信息
echo "install prefix            $prefix"
echo "source path               $source_path"
echo "C compiler                $cc"
echo "ARCH                      $arch ($cpu)"
if test "$build_suffix" != ""; then
    echo "build suffix              $build_suffix"
fi
if test "$progs_suffix" != ""; then
    echo "progs suffix              $progs_suffix"
fi
if test "$extra_version" != ""; then
    echo "version string suffix     $extra_version"
fi
#${}的特异功能：
#${file-my.file.txt}假如 $file 为空值，则使用 my.file.txt 作默认值。(保留没设定及非空值)
#在这里，如果某个变量为空值，则取默认值为no
echo "big-endian                ${bigendian-no}"
echo "runtime cpu detection     ${runtime_cpudetect-no}"
if enabled x86; then
    echo "${yasmexe}                      ${yasm-no}"
    echo "MMX enabled               ${mmx-no}"
    echo "MMX2 enabled              ${mmx2-no}"
    echo "3DNow! enabled            ${amd3dnow-no}"
    echo "3DNow! extended enabled   ${amd3dnowext-no}"
    echo "SSE enabled               ${sse-no}"
    echo "SSSE3 enabled             ${ssse3-no}"
    echo "AVX enabled               ${avx-no}"
    echo "CMOV enabled              ${cmov-no}"
    echo "CMOV is fast              ${fast_cmov-no}"
    echo "EBX available             ${ebx_available-no}"
    echo "EBP available             ${ebp_available-no}"
fi
if enabled arm; then
    echo "ARMv5TE enabled           ${armv5te-no}"
    echo "ARMv6 enabled             ${armv6-no}"
    echo "ARMv6T2 enabled           ${armv6t2-no}"
    echo "ARM VFP enabled           ${armvfp-no}"
    echo "IWMMXT enabled            ${iwmmxt-no}"
    echo "NEON enabled              ${neon-no}"
fi
if enabled mips; then
    echo "MMI enabled               ${mmi-no}"
fi
if enabled ppc; then
    echo "AltiVec enabled           ${altivec-no}"
    echo "PPC 4xx optimizations     ${ppc4xx-no}"
    echo "dcbzl available           ${dcbzl-no}"
fi
if enabled sparc; then
    echo "VIS enabled               ${vis-no}"
fi
echo "debug symbols             ${debug-no}"
echo "strip symbols             ${stripping-no}"
echo "optimize for size         ${small-no}"
echo "optimizations             ${optimizations-no}"
echo "static                    ${static-no}"
echo "shared                    ${shared-no}"
echo "postprocessing support    ${postproc-no}"
echo "new filter support        ${avfilter-no}"
```

```
echo "network support           ${network-no}"
echo "threading support         ${thread_type-no}"
echo "safe bitstream reader     ${safe_bitstream_reader-no}"
echo "SDL support               ${sdl-no}"
echo "Sun medialib support      ${mlib-no}"
echo "libdxva2 enabled          ${dxva2-no}"
echo "libva enabled             ${vaapi-no}"
echo "libvdpau enabled          ${vdpau-no}"
echo "AVISynth enabled          ${avisynth-no}"
echo "frei0r enabled            ${frei0r-no}"
echo "gnutls enabled            ${gnutls-no}"
echo "libaacplus enabled        ${libaacplus-no}"
echo "libass enabled            ${libass-no}"
echo "libcdio support           ${libcdio-no}"
echo "libcelt enabled           ${libcelt-no}"
echo "libdc1394 support         ${libdc1394-no}"
echo "libdirac enabled          ${libdirac-no}"
echo "libfaac enabled           ${libfaac-no}"
echo "libgsm enabled            ${libgsm-no}"
echo "libmodplug enabled        ${libmodplug-no}"
echo "libmp3lame enabled        ${libmp3lame-no}"
echo "libnut enabled            ${libnut-no}"
echo "libopencore-amrnb support ${libopencore_amrnb-no}"
echo "libopencore-amrwb support ${libopencore_amrwb-no}"
echo "libopencv support         ${libopencv-no}"
echo "libopenjpeg enabled       ${libopenjpeg-no}"
echo "libpulse enabled          ${libpulse-no}"
echo "librtmp enabled           ${librtmp-no}"
echo "libschroedinger enabled   ${libschroedinger-no}"
echo "libspeex enabled          ${libspeex-no}"
echo "libstagefright-h264 enabled    ${libstagefright_h264-no}"
echo "libtheora enabled         ${libtheora-no}"
echo "libutvideo enabled        ${libutvideo-no}"
echo "libv4l2 enabled           ${libv4l2-no}"
echo "libvo-aacenc support      ${libvo_aacenc-no}"
echo "libvo-amrwbenc support    ${libvo_amrwbenc-no}"
echo "libvorbis enabled         ${libvorbis-no}"
echo "libvpx enabled            ${libvpx-no}"
echo "libx264 enabled           ${libx264-no}"
echo "libxavs enabled           ${libxavs-no}"
echo "libxvid enabled           ${libxvid-no}"
echo "openal enabled            ${openal-no}"
echo "openssl enabled           ${openssl-no}"
echo "zlib enabled              ${zlib-no}"
echo "bzlib enabled             ${bzlib-no}"
echo

for type in decoder encoder hwaccel parser demuxer muxer protocol filter bsf indev outdev; do
    echo "Enabled ${type}s:"
    eval list=\$$(toupper $type)_LIST
    print_enabled '_*' $list | sort | pr -r -3 -t
    echo
done

license="LGPL version 2.1 or later"
if enabled nonfree; then
    license="nonfree and unredistributable"
elif enabled gplv3; then
    license="GPL version 3 or later"
elif enabled lgplv3; then
    license="LGPL version 3 or later"
elif enabled gpl; then
    license="GPL version 2 or later"
fi

echo "License: $license"
#创建config.mak和config.h
#根据情况也会创建config.asm
echo "Creating config.mak and config.h..."

test -e Makefile || $ln_s "$source_path/Makefile" .

enabled stripping || strip="echo skipping strip"
#重要：需要输出的文件
#TMPH是一个临时文件，最终会拷贝给config.h
config_files="$TMPH config.mak"
#写入config.mak文件
#首先写入一些基本信息
#"<<EOF"表示后续的输入作为子命令或子shell的输入，直到遇到"EOF"，再次返回到
#主调shell，可将其理解为分界符（delimiter）。
#最后的"EOF"必须单独占一行
cat > config.mak <<EOF
# Automatically generated by configure - do not modify!
ifndef FFMPEG_CONFIG_MAK
FFMPEG_CONFIG_MAK=1
FFMPEG_CONFIGURATION=$FFMPEG_CONFIGURATION
prefix=$prefix
LIBDIR=\$(DESTDIR)$libdir
SHLIBDIR=\$(DESTDIR)$shlibdir
INCDIR=\$(DESTDIR)$incdir
BINDIR=\$(DESTDIR)$bindir
```

```
DATADIR=\$(DESTDIR)$datadir
MANDIR=\$(DESTDIR)$mandir
SRC_PATH=$source_path
ifndef MAIN_MAKEFILE
SRC_PATH:=\$(SRC_PATH:.%=..%)
endif
CC_IDENT=$cc_ident
ARCH=$arch
CC=$cc
CXX=$cxx
AS=$as
LD=$ld
DEPCC=$dep_cc
YASM=$yasmexe
YASMDEP=$yasmexe
AR=$ar
RANLIB=$ranlib
CP=cp -p
LN_S=$ln_s
STRIP=$strip
CPPFLAGS=$CPPFLAGS
CFLAGS=$CFLAGS
CXXFLAGS=$CXXFLAGS
ASFLAGS=$ASFLAGS
AS_O=$CC_O
CC_O=$CC_O
CXX_O=$CXX_O
LDFLAGS=$LDFLAGS
FFSERVERLDFLAGS=$FFSERVERLDFLAGS
SHFLAGS=$SHFLAGS
YASMFLAGS=$YASMFLAGS
BUILDSUF=$build_suffix
PROGSSUF=$progs_suffix
FULLNAME=$FULLNAME
LIBPREF=$LIBPREF
LIBSUF=$LIBSUF
LIBNAME=$LIBNAME
SLIBPREF=$SLIBPREF
SLIBSUF=$SLIBSUF
EXESUF=$EXESUF
EXTRA_VERSION=$extra_version
DEPFLAGS=$DEPFLAGS
CCDEP=$CCDEP
CXXDEP=$CXXDEP
ASDEP=$ASDEP
CC_DEPFLAGS=$CC_DEPFLAGS
AS_DEPFLAGS=$AS_DEPFLAGS
HOSTCC=$host_cc
HOSTCFLAGS=$host_cflags
HOSTEXESUF=$HOSTEXESUF
HOSTLDFLAGS=$host_ldflags
HOSTLIBS=$host_libs
TARGET_EXEC=$target_exec
TARGET_PATH=$target_path
SDL_LIBS=$sdl_libs
SDL_CFLAGS=$sdl_cflags
LIB_INSTALL_EXTRA_CMD=$LIB_INSTALL_EXTRA_CMD
EXTRALIBS=$extralibs
INSTALL=$install
LIBTARGET=${LIBTARGET}
SLIBNAME=${SLIBNAME}
SLIBNAME_WITH_VERSION=${SLIBNAME_WITH_VERSION}
SLIBNAME_WITH_MAJOR=${SLIBNAME_WITH_MAJOR}
SLIB_CREATE_DEF_CMD=${SLIB_CREATE_DEF_CMD}
SLIB_EXTRA_CMD=${SLIB_EXTRA_CMD}
SLIB_INSTALL_NAME=${SLIB_INSTALL_NAME}
SLIB_INSTALL_LINKS=${SLIB_INSTALL_LINKS}
SLIB_INSTALL_EXTRA_LIB=${SLIB_INSTALL_EXTRA_LIB}
SLIB_INSTALL_EXTRA_SHLIB=${SLIB_INSTALL_EXTRA_SHLIB}
SAMPLES:=${samples:-\$(FATE_SAMPLES)}
NOREDZONE_FLAGS=$noredzone_flags
EOF
#获取版本
#主要通过各个类库文件夹中的version.h文件
#读取XXX_VERSION（相当于把头文件当成一个文本来读）
get_version(){
    name=$1
    file=$source_path/$2
# This condition will be removed when we stop supporting old libpostproc versions
if ! test "$name" = LIBPOSTPROC || test "$postproc_version" = current; then
    eval $(grep "#define ${name}_VERSION_M" "$file" | awk '{ print $2"="$3 }')
    eval ${name}_VERSION=\$${name}_VERSION_MAJOR.\$${name}_VERSION_MINOR.\$${name}_VERSION_MICRO
fi
    lcname=$(tolower $name)
    eval echo "${lcname}_VERSION=\$${name}_VERSION" >> config.mak
    eval echo "${lcname}_VERSION_MAJOR=\$${name}_VERSION_MAJOR" >> config.mak
}
#获取版本
get_version LIBAVCODEC  libavcodec/version.h
get_version LIBAVDEVICE libavdevice/avdevice.h
get_version LIBAVFILTER libavfilter/version.h
```

```
get_version LIBAVFORMAT libavformat/version.h
get_version LIBAVUTIL   libavutil/avutil.h
get_version LIBPOSTPROC libpostproc/postprocess.h
get_version LIBSWRESAMPLE libswresample/swresample.h
get_version LIBSWSCALE  libswscale/swscale.h
#config.h前面需要添加的一些内容（TMPH是一个临时文件，最终会拷贝给config.h）
cat > $TMPH <<EOF
/* Automatically generated by configure - do not modify! */
#ifndef FFMPEG_CONFIG_H
#define FFMPEG_CONFIG_H
#define FFMPEG_CONFIGURATION "$(c_escape $FFMPEG_CONFIGURATION)"
#define FFMPEG_LICENSE "$(c_escape $license)"
#define FFMPEG_DATADIR "$(eval c_escape $datadir)"
#define AVCONV_DATADIR "$(eval c_escape $datadir)"
#define CC_TYPE "$cc_type"
#define CC_VERSION $cc_version
#define restrict $_restrict
#define EXTERN_PREFIX "${extern_prefix}"
#define EXTERN_ASM ${extern_prefix}
#define SLIBSUF "$SLIBSUF"
EOF

test -n "$malloc_prefix" &&
    echo "#define MALLOC_PREFIX $malloc_prefix" >>$TMPH

if enabled small || disabled optimizations; then
    echo "#undef  av_always_inline"  >> $TMPH
    if enabled small; then
        echo "#define av_always_inline inline"  >> $TMPH
    else
        echo "#define av_always_inline av_unused"  >> $TMPH
    fi
fi
#包含yasm
if enabled yasm; then
    append config_files $TMPASM
    printf '' >$TMPASM
fi
#输出所有的配置信息包含3类：
#以“ARCH_”开头，包含系统架构信息
#以“HAVE_”开头，包含系统特征信息
#以“CONFIG_”开头，包含编译配置（数量最多，包含协议、复用器、编解码器等的配置，将近1000行）
#config_files
print_config ARCH_   "$config_files" $ARCH_LIST
print_config HAVE_   "$config_files" $HAVE_LIST
print_config CONFIG_ "$config_files" $CONFIG_LIST       \
                                     $CONFIG_EXTRA      \
                                     $ALL_COMPONENTS    \
#经过测试的组件？
cat >>config.mak <<EOF
ACODEC_TESTS=$(print_enabled -n _test $ACODEC_TESTS)
VCODEC_TESTS=$(print_enabled -n _test $VCODEC_TESTS)
LAVF_TESTS=$(print_enabled   -n _test $LAVF_TESTS)
LAVFI_TESTS=$(print_enabled  -n _test $LAVFI_TESTS)
SEEK_TESTS=$(print_enabled   -n _test $SEEK_TESTS)
EOF

echo "#endif /* FFMPEG_CONFIG_H */" >> $TMPH
#结束了
echo "endif # FFMPEG_CONFIG_MAK" >> config.mak

# 关键：临时文件拷贝至config.h
# Do not overwrite an unchanged config.h to avoid superfluous rebuilds.
# 配置没有变化的时候，不重新生成config.h（重新生成config.h会导致大量文件需要重新编译）
#
cp_if_changed $TMPH config.h
# touch fileA
# 如果fileA不存在，touch指令会在当前目录下新建一个空白文件fileA。
touch .config

enabled yasm && cp_if_changed $TMPASM config.asm

cat > $TMPH <<EOF
/* Generated by ffconf */
#ifndef AVUTIL_AVCONFIG_H
#define AVUTIL_AVCONFIG_H
EOF

test "$postproc_version" != current && cat >> $TMPH <<EOF
#define LIBPOSTPROC_VERSION_MAJOR $LIBPOSTPROC_VERSION_MAJOR
#define LIBPOSTPROC_VERSION_MINOR $LIBPOSTPROC_VERSION_MINOR
#define LIBPOSTPROC_VERSION_MICRO $LIBPOSTPROC_VERSION_MICRO
EOF

print_config AV_HAVE_ $TMPH $HAVE_LIST_PUB

echo "#endif /* AVUTIL_AVCONFIG_H */" >> $TMPH

cp_if_changed $TMPH libavutil/avconfig.h

test -n "$WARNINGS" && printf "\n$WARNINGS"
```

```
3850.
3851.  # build pkg-config files
3852.
3853.  pkgconfig_generate(){
3854.  name=$1
3855.  shortname=${name#lib}${build_suffix}
3856.  comment=$2
3857.  version=$3
3858.  libs=$4
3859.  requires=$5
3860.  enabled ${name#lib} || return 0
3861.  mkdir -p $name
3862.  cat <<EOF > $name/$name.pc
3863.  prefix=$prefix
3864.  exec_prefix=\${prefix}
3865.  libdir=$libdir
3866.  includedir=$incdir
3867.
3868.  Name: $name
3869.  Description: $comment
3870.  Version: $version
3871.  Requires: $(enabled shared || echo $requires)
3872.  Requires.private: $(enabled shared && echo $requires)
3873.  Conflicts:
3874.  Libs: -L\${libdir} -l${shortname} $(enabled shared || echo $libs)
3875.  Libs.private: $(enabled shared && echo $libs)
3876.  Cflags: -I\${includedir}
3877.  EOF
3878.  cat <<EOF > $name/$name-uninstalled.pc
3879.  prefix=
3880.  exec_prefix=
3881.  libdir=\${pcfiledir}
3882.  includedir=${source_path}
3883.
3884.  Name: $name
3885.  Description: $comment
3886.  Version: $version
3887.  Requires: $requires
3888.  Conflicts:
3889.  Libs: \${libdir}/${LIBPREF}${shortname}${LIBSUF} $libs
3890.  Cflags: -I\${includedir}
3891.  EOF
3892.  }
3893.
3894.  pkgconfig_generate libavutil "FFmpeg utility library" "$LIBAVUTIL_VERSION" "$LIBM"
3895.  pkgconfig_generate libavcodec "FFmpeg codec library" "$LIBAVCODEC_VERSION" "$extralibs" "libavutil = $LIBAVUTIL_VERSION"
3896.  pkgconfig_generate libavformat "FFmpeg container format library" "$LIBAVFORMAT_VERSION" "$extralibs" "libavcodec = $LIBAVCODEC_VERSIO
3897.  pkgconfig_generate libavdevice "FFmpeg device handling library" "$LIBAVDEVICE_VERSION" "$extralibs" "libavformat = $LIBAVFORMAT_VERSI
3898.  pkgconfig_generate libavfilter "FFmpeg video filtering library" "$LIBAVFILTER_VERSION" "$extralibs"
3899.  pkgconfig_generate libpostproc "FFmpeg postprocessing library" "$LIBPOSTPROC_VERSION" "" "libavutil = $LIBAVUTIL_VERSION"
3900.  pkgconfig_generate libswscale "FFmpeg image rescaling library" "$LIBSWSCALE_VERSION" "$LIBM" "libavutil = $LIBAVUTIL_VERSION"
3901.  pkgconfig_generate libswresample "FFmpeg audio rescaling library" "$LIBSWRESAMPLE_VERSION" "$LIBM" "libavutil = $LIBAVUTIL_VERSION"
```

雷霄骅
leixiaohua1020@126.com
http://blog.csdn.net/leixiaohua1020

文章标签： FFmpeg Configure Shell 源代码 Make

个人分类： FFMPEG

所属专栏： FFmpeg