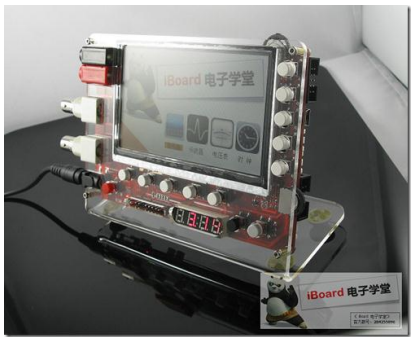
《iBoard 电子学堂》第四讲
任意波发生器软件架构

关于 《iBoard 电子学堂》……



《iBoard 电子学堂》是一个综合型的电子研发开发平台，适合在校学生、一线工程师及电子爱好者等。

交流方式：

*官方博客：*XiaomaGee.cnblogs.com

*官方论坛：*www.oshcn.com

*官方淘宝店铺：*i-Board.taobao.com

QQ 群：

《iBoard 电子学堂 群【A】》：204255896（500 人，已满）

《iBoard 电子学堂 群【B】》：165201798（500 人超级群）

《iBoard 电子学堂 群【C】》：215053598（200 人高级群）

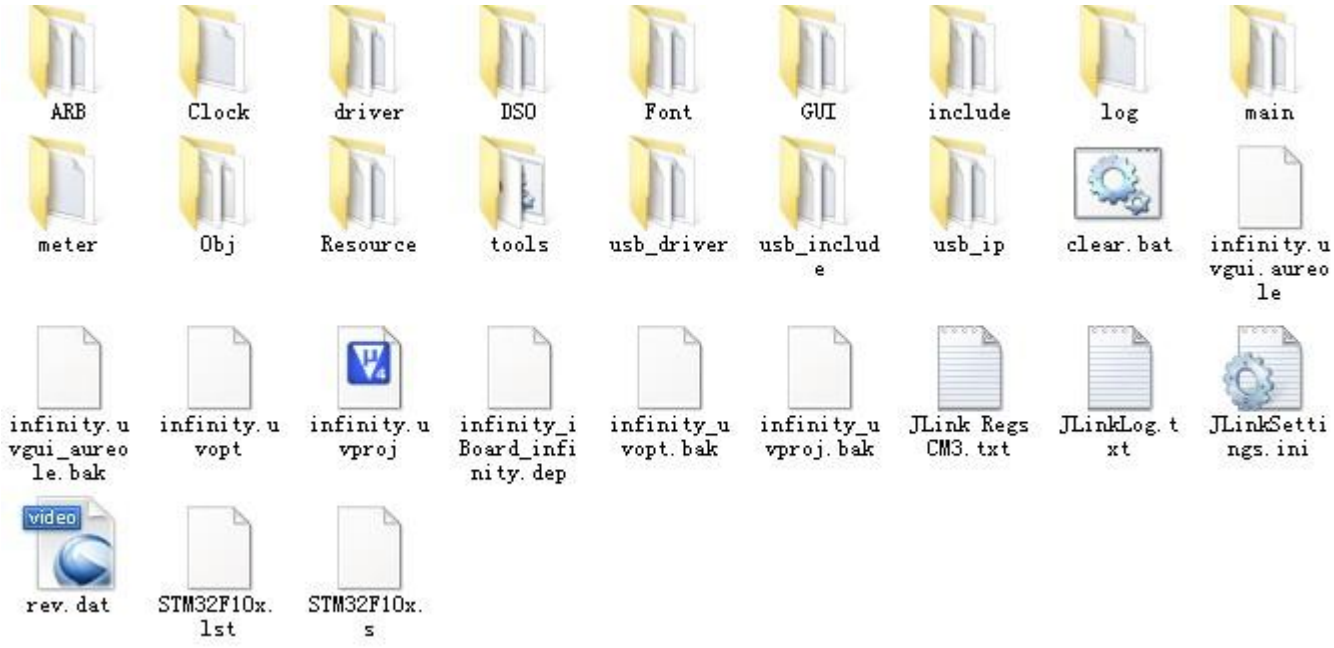
《iBoard 电子学堂 群【D】》：215054675（200 人高级群）

《iBoard 电子学堂 群【E】》： 215055211（200 人高级群）

《iBoard 电子学堂 群【F】》： 78538605 （200 人高级群）

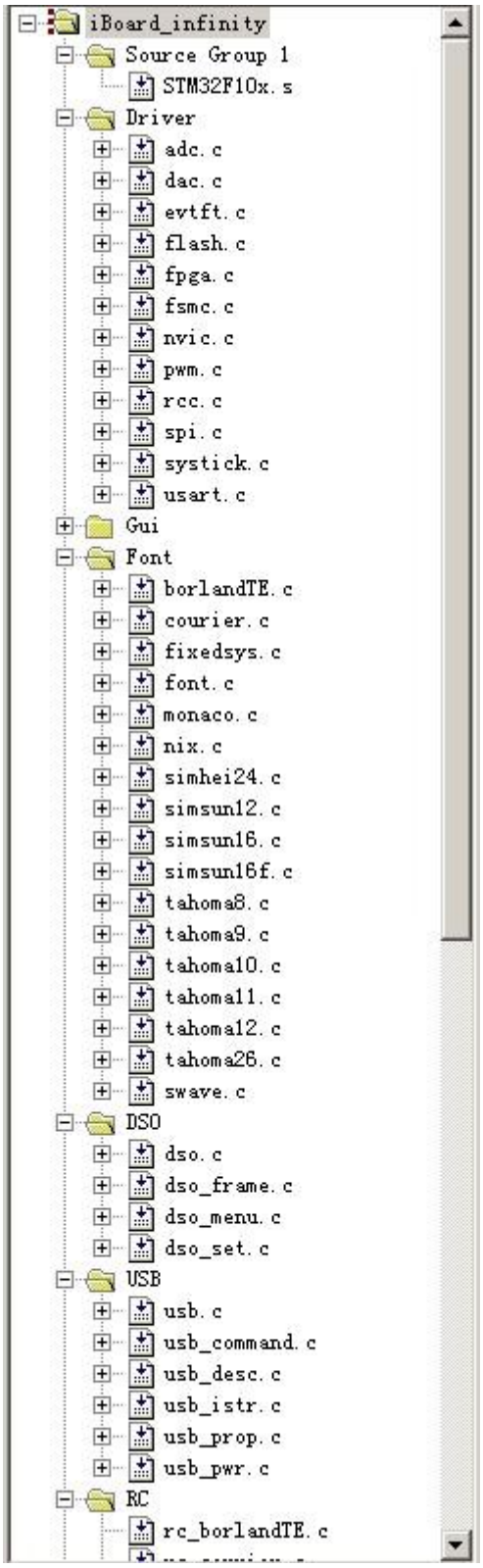
开始：

王紫豪-XiaomaGee(15959622)　20:27:41
群课开始了，这节课我讲《iBoard　电子学堂》任意波发生器软件架构。由于是三群直播，所以可能会有点慢，大家谅解。这也要求讲课期间大家不要说话，不然乱套了
王紫豪-XiaomaGee(15959622)　20:28:31
为了区别，我使用红色字体，请大家谅解
首先，请大家去下载今天课程所讲的软件包，下载地址如下：
http://files.cnblogs.com/xiaomagee/infinity_2012.4.6.7z
王紫豪-XiaomaGee(15959622)　20:31:33
对的，全部代码
下载好的同学　，请放到群共享里让大家下载
王紫豪-XiaomaGee(15959622)　20:33:28
请大家去群共享下载
王紫豪-XiaomaGee(15959622)　20:35:03
趁大家的下载时间，我来回顾下上一节课的内容
王紫豪-XiaomaGee(15959622)　20:39:08
里面包含了四个文件夹，和三个说明文件；其中：8051　是　51 单片机的源代码，fpga 里面是 fpga 逻辑固件和 niosii 代码，stm32 文件夹内为　stm32 的源代码，烧写文件内是编译好的二进制，可以直接烧写。
王紫豪-XiaomaGee(15959622)　20:40:22
今天，我们的目标是　stm32 里面的内容。大家可以打开文件夹看一下，里面是一个完整的　KEIL　MDK　工程，包含了若干文件夹



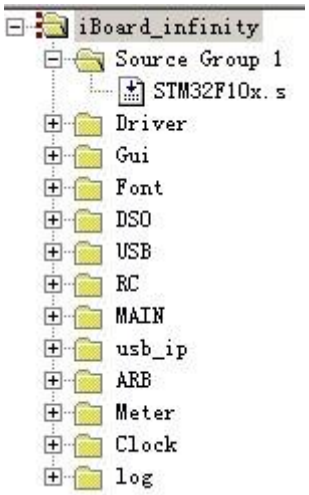
王紫豪-XiaomaGee(15959622)　20:41:37
大家不要小看这个文件包，它包含了上万行的源代码（没有完整统计），大大小小几十个　c　文件。我把他称为《iBoard　infinity》固件包。

电脑里安装 keil mdk 的同学，可以打开工程，没有安装 KEIL MDK 也没关系，c 源文件是文本文件，使用最简单的记事本就可以打开
王紫豪-XiaomaGee(15959622) 20:43:37
打开后，在工程管理的栏目里，我们可以看到所有的源文件

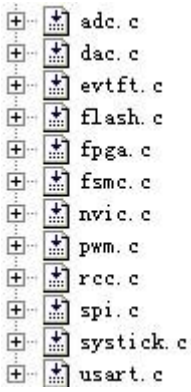


这个截屏，还不是全部。

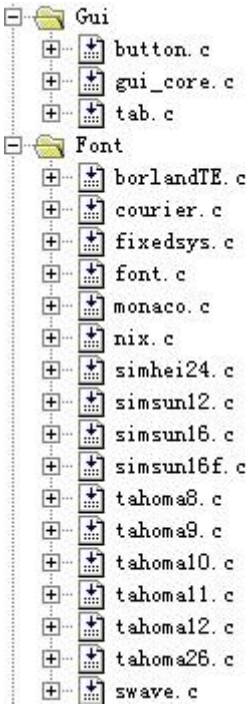
我们把它折叠起来，这样就简单明了了。



王紫豪-XiaomaGee(15959622)　20:44:45
为了讲课需要，我把各个文件夹的功能大致说一下
首先，自上而下，stm32f10x.s 是个启动代码，以 s 为扩展名的是汇编代码。
王紫豪-XiaomaGee(15959622)　20:46:01
driver　文件夹，为驱动文件，包含了 cpu 外设以及外部芯片的最底层驱动

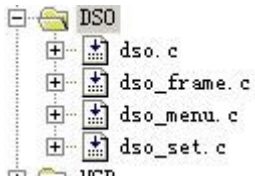


王紫豪-XiaomaGee(15959622)　20:47:28
包含了 adc，dac，液晶驱动，flash，fpga，总线..等等的驱动；这里说一下，我没采用 stm32 的系统库，而是直接操作寄存器的；这些驱动都是封装好的，大家需要的话，直接用即可
王紫豪-XiaomaGee(15959622)　20:48:32
GUI 和 FONT　文件夹为　X-GUI　源代码以及字库驱动，它包含了最底层的绘图操作



王紫豪-XiaomaGee(15959622)　20:49:34

大家可以看到，字库驱动较多；这也是图形界面做得漂亮的一个前提。关于 X-GUI ，以后的课程会做详细的讲解；

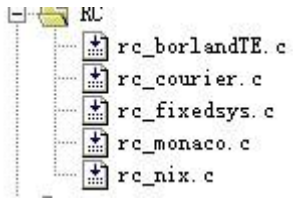


这个是 DSO 文件夹，是《数字存储示波器》的简称，所以，本文件夹内就是示波器的实现；今天也不讲这个

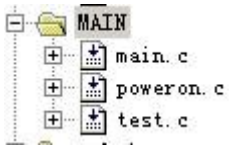
王紫豪-XiaomaGee(15959622)　20:51:14

USB 文件夹和 USB_IP 文件夹，是 stm32 的 usb 驱动，我们配合 上位机软件，可以烧写字库以及截屏等等。具体的请参考我的博文

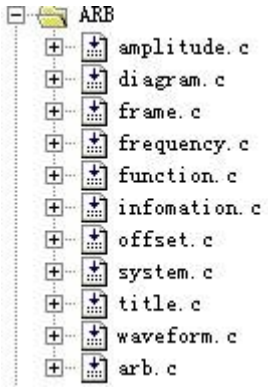
RC 文件夹是一些资源文件，主要是一些英文字体字库



王紫豪-XiaomaGee(15959622)　20:52:44



main 文件夹就是我们得 main 文件和开机启动界面的代码。



ARB 文件夹，就是我们今天主讲的内容，ARB 是《任意波形发生器》的英文简写

王紫豪-XiaomaGee(15959622) 20:54:03

Meter 文件夹内，为 500V 交直流电压表的驱动，它包含了自动量程算法。这个以后的课程会讲解

Clock 文件夹内，是时钟显示界面的实现，这个很简单。

王紫豪-XiaomaGee(15959622) 20:55:15

至此，文件介绍就说到这里。

下面我们切入正题，讲解任意波形发生器的代码。

王紫豪-XiaomaGee(15959622) 20:57:28

笼统地讲，波形发生器功能包含了两部分：

第一：人机界面部分

第二：底层驱动部分

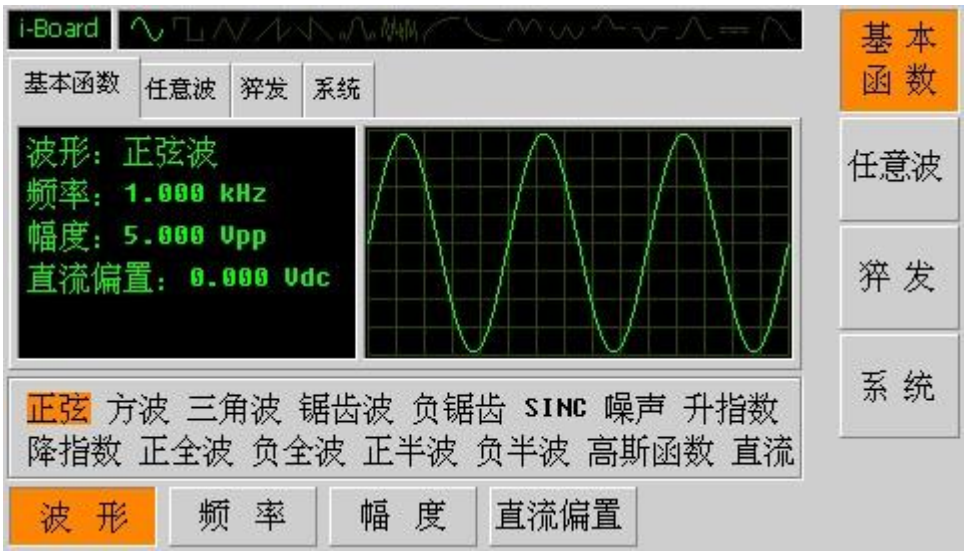
王紫豪-XiaomaGee(15959622) 20:58:55

①人机界面部分，负责监控按键操作，并根据用户意愿，完成图形显示。

②底层驱动部分，是通过操作 fpga 或者其他资源，来完成波形参数的设置。

王紫豪-XiaomaGee(15959622) 21:00:40

我首先介绍下主界面，下图是主界面的截图



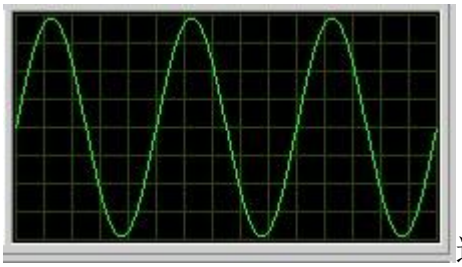
图片模糊的网友，可以把聊天窗口最大化，这样 1:1 显示清晰

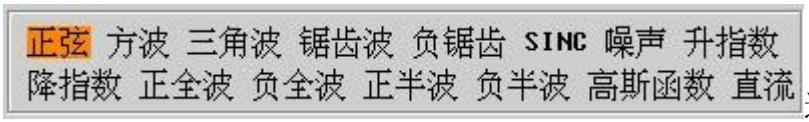
王紫豪-XiaomaGee(15959622) 21:02:10

这个界面使用 X-GUI 实现的一个简单界面，它包含了下面几部分：

最上面是波形索引： 

这里是波形参数显示。


这里是输出波形示意图



这里是设置栏

王紫豪-XiaomaGee(15959622) 21:03:28

最下面是虚拟按键

我再切换几个别的界面给大家看看。下图是设置三角波对称度的页面

王紫豪-XiaomaGee(15959622) 21:05:03

实际工作过程中，再参数显示那里，对称度是闪烁的。我们这里是静态的图片，看不到。

大家把目光转移到源代码这里，首先我们从 main 函数说起。

main 函数再 main 文件夹内。

王紫豪-XiaomaGee(15959622)　21:07:17

main　函数包含了若干头文件，这些具体是什么内容大家可以先不用管。毕竟这个工程很庞大

```
//-----------------Include files-------------------------//
#include "..\include\nvic.h"
#include "..\include\rcc.h"
#include "..\include\dac.h"
#include "..\include\usart.h"
#include "..\include\adc.h"
#include "..\include\hardware.h"
#include "..\include\systick.h"
#include "..\include\evtft.h"
#include "..\include\pwm.h"
#include "..\include\fpga.h"
#include "..\include\gui_core.h"
#include "..\include\font.h"
#include "..\include\spi.h"
#include "..\include\flash.h"
#include "..\usb_include\usb.h"
#include "..\usb_include\usb_command.h"
#include "..\include\event.h"
#include "..\include\arb.h"
#include "..\include\main.h"
#include "..\include\dso.h"
#include "..\include\meter.h"
#include "..\include\power_on.h"


#include <string.h>
#include <stdlib.h>
#include <math.h>
```

我们找到　main　函数

王紫豪-XiaomaGee(15959622)　21:09:24

```
  power_on();
  arb.initialize();
```

power_on 是开机以及初始化，完成后，进入如下界面：

王紫豪-XiaomaGee(15959622)　　21:10:29
然后我们通过方向键和确认键，去选择信号源功能。这里，我使用函数指针，来进入这四个功能的一个。

函数指针定义如下：

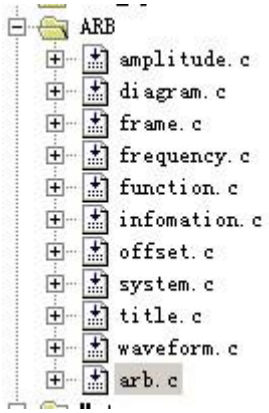
```
int(*fun[4]) (void) = {
    arb.main,
    dso.main,
    meter.main,
    clock_main
};
```

他的执行，是在这里：

```
if (event == KEY_ENTER) {
    for (i = 0; i <= 100; i++) {
        pwm.initialize(100 - i);
        for (j = 0; j < 20000; j++) ;
    }

    fun[main_fun]();
    first_run = 1;
    event = 0;
    flag = 1;
    show_logo();
}
```

王紫豪-XiaomaGee(15959622)　　21:11:30
这一段表达的意思，是通过按下《确认》键，来选择相应的功能。

这里一下子就跳到了，我们 arb 文件夹内的 arb.main 函数里



在这里！！

王紫豪-XiaomaGee(15959622) 21:13:55

大家打开 arb.c 文件。

从上面看到

```
const ARB_T arb = {
    .main = arb_main,
    .initialize=initialize
};
```

这里，我使用了 C99 的赋值方式。

《iBoard infinity》固件包内，所有的类型定义，都在相对应的 .h 头文件里。

王紫豪-XiaomaGee(15959622) 21:15:14

假如需要找 ARB_T 的定义，我们就去找 arb.h 里面；翻开后，类型定义如下：

```
typedef const struct{
    int (* main)(void);
    int (* initialize)(void);
}ARB_T;
```

王紫豪-XiaomaGee(15959622) 21:16:20

我这样做了一个简单的封装，是为了使用方便。并且保证代码的整洁，整个工程

使用了类似的模式。如：

```
37
38  typedef struct{
39    unsigned long int (* read_jedec_id)(void);
40    unsigned char (* read)(unsigned long int /* address */);
41    int (* read_32)(unsigned long int /* addr */, char * /* buf */);
42    int (* read_72)(unsigned long int /* addr */, char * /* buf */);
43    int (* read_256)(unsigned long int /* addr */, char * /* buf */);
44
45    int (* write)(unsigned long int /* address */, char /* dat */);
46    int (* write_256)(unsigned long int /* address */, char * /* buf */);
47
48    int (* chip_erase)(void);
49    int (* lock)(void);
50    int (* unlock)(void);
51    int (* sector_erase)(int sector);
52  }SFLASH_T;
53
```

王紫豪-XiaomaGee(15959622)　　21:17:20

下面接着分析　arb.c

细心的网友可以发现，arb.c 仅仅包含了 arb_main 这一个函数；

王紫豪-XiaomaGee(15959622)　　21:19:16

其实，我们是以功能为划分依据，把他们分类的。这个功能体现在主界面的　tab 上。

现在看来仅仅有四个功能，后期会扩充更多的功能，如：调制、扫频、键控 等等功能。这些功能硬件上都是支持的，只需要增加代码既可。

王紫豪-XiaomaGee(15959622)　　21:21:12

从　arb_main　函数内，我们默认跳入了　function.c 里，这个文件实现了【基本函数】的人机界面和参数设置

王紫豪-XiaomaGee(15959622)　　21:22:17

我们现在进入　function.C 里，　然后找到函数

static　int　function_main(void)

细心的网友会发现，　我再大部分函数前面都增加了　static　修饰词

王紫豪-XiaomaGee(15959622)　　21:23:43

这样配合函数指针的重定义方式，　来实现各个函数内的函数名可以重复的功能.　增加了代码的可靠性.

王紫豪-XiaomaGee(15959622) 21:25:12

```
static int
function_main(void)
{
  int i;
  int(*fun_function[4][5]) (void) = {
    { //sine
      _waveform,
      _frequency,
      _amplitude,
      _offset,
      NULL
    }, { //squ
      _waveform,
      _frequency,
      _amplitude,
      _offset,
      duty_cycle
    }, { //tri
      _waveform,
      _frequency,
      _amplitude,
      _offset,
      symmetry
    }, {
      _waveform,
      _offset,
      NULL,
      NULL,
      NULL
    }
  };
```

function_main 函数的开始,又是一堆函数指针.....没错, 整个工程就是这样的模式, 这样可以改善功能添加/删减所带来的工作量

课间休息 :10 分钟 ~~~~~~
王紫豪-XiaomaGee(15959622) 21:36:58
上几次没听课的同学,去这里下载群课笔记:

- 《iBoard 电子学堂》 群课第一课笔记：嵌入式系统电源设计
- 《iBoard 电子学堂》 群课第二课笔记：iBoard 原理图解析
- 《iBoard 电子学堂》 群课第三课笔记：任意波发生器电路详解

http://www.oshcn.com/thread-13823-1-1.html

王紫豪-XiaomaGee(15959622) 21:39:50
接着讲课了。课间总是很短。。。。。。

王紫豪-XiaomaGee(15959622)　21:42:05
刚才讲到了 function_main 这个函数，首先看到的是函数指针定义：

```
int(*fun_function[4][5]) (void) = {
  { //sine
    _waveform,
    _frequency,
    _amplitude,
    _offset,
    NULL
  }, { //squ
    _waveform,
    _frequency,
    _amplitude,
    _offset,
    duty_cycle
  }, { //tri
    _waveform,
    _frequency,
    _amplitude,
    _offset,
    symmetry
  }, {
    _waveform,
    _offset,
    NULL,
    NULL,
    NULL
  }
};
```

王紫豪-XiaomaGee(15959622)　21:43:07
其实，这个是对应菜单功能的，菜单内容在全局变量里。也就是这个函数的上面：

```
]char const *  button_function[/*function */ 4][/* language */ 2][5]
]  { //common
]    {
       "  波  形  ",
       "  频  率  ",
       "  幅  度  ",
       "直流偏置",
       NULL
     }, {
       " Waveform",
       " Frequency",
       " Amplitude",
       " DC Offset",
       NULL
     }
   }, {     //SQU
]    {
       "  波  形  ",
       "  频  率  ",
       "  幅  度  ",
       "直流偏置",
       "  占空比 "
     }, {
       " Waveform",
       " Frequency",
       " Amplitude",
       " DC Offset",
       " Duty Cycle"
     }
   }, {     //TRI
```

大家可以看出，这个是中英文界面，我们通过一个全局变量来选择的。

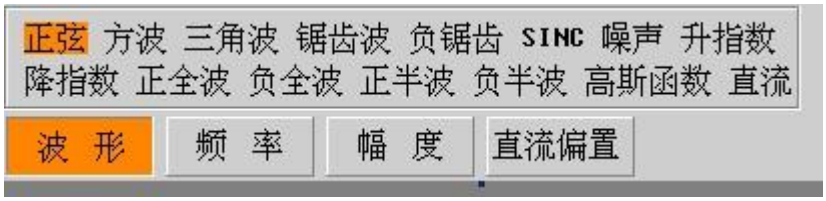
王紫豪-XiaomaGee(15959622) 21:44:16

往下走，第 206 行开始，是函数

static int _waveform(void)

王紫豪-XiaomaGee(15959622) 21:45:25

它主要实现了这个界面：



也就是波形选择，这里有 n 个波形，呵呵。。。

```
const char * wave_index[2] = {
  " 正弦 方波 三角波 锯齿波 负锯齿 SINC 噪声 升指数\n 降指数 正全波
  " SIN SQU TRI RAMP N_RAMP SINC Noise EXP_R\n EXP_F FULL_P FULL_N
};
```

定义在这里

王紫豪-XiaomaGee(15959622) 21:48:08

代码里很多是关于 X-GUI 内容的，其实也很简单，例如：

```
s.x = 4;
s.y = 193;
s.space.line = 5;
s.space.word = 0;
s.background_color = COLOR_WINDOW_BACKGROUND;

s.color = COLOR_BLACK;

p = wave_index[sys.language];
```

这个是一个字符串，通过 x-gui 显示再界面上。
s.x 是 x 坐标；y 是 y 坐标，下面两个分别为前景色和背景色。当然还有字体设置：

```
font._default.single_byte = &fixedsys;
```

整个函数就是通过查询用户事件，来驱动功能设定个。
王紫豪-XiaomaGee(15959622) 21:49:55

```
if (
  event == KEY_F2 ||
  event == KEY_F3 ||
  event == KEY_F4 ||
  event == KEY_F5
  ) break;
if (
  event == KEY_M1 ||
  event == KEY_M2 ||
  event == KEY_M3 ||
  event == KEY_M4 ||
  event == KEY_M5 ||
  event == KEY_ESC
  ) break;

if (event == KEY_LEFT || event == KEY_UP) {
  event = EVENT_CLEAR;
  position--;
```

这里一直判断 event 变量，event 变量通过后台设定的
王紫豪-XiaomaGee(15959622) 21:51:40
下面的时间，我说下底层驱动。
王紫豪-XiaomaGee(15959622) 21:52:46
frequency.c 是频率设置函数

```
static int
set(double freq)
{
  double temp = 281474976710656. / 100000000.;
  unsigned long long int fword;

  if (freq > 10000000. || freq < 0.) return -1;

  temp *= freq;
  fword = temp;

  fpga_ctl.b.ctl1 = 0;
  fpga_ctl.b.ctl2 = 0;
  fpga_ctl.b.ctl3 = 0;
  fpga.ctl();

  FPGA_WR2 = (fword >> 32) & 0xffff;

  fpga_ctl.b.ctl1 = 1;
  fpga_ctl.b.ctl2 = 0;
  fpga_ctl.b.ctl3 = 0;
  fpga.ctl();

  FPGA_WR2 = (fword >> 16) & 0xffff;

  fpga_ctl.b.ctl1 = 0;
  fpga_ctl.b.ctl2 = 1;
  fpga_ctl.b.ctl3 = 0;
  fpga.ctl();

  FPGA_WR2 = fword & 0xffff;

  return 0;
}
```

大家可能看到一个比较大的数字

```
281474976710656. / 100000000.;
```

王紫豪-XiaomaGee(15959622) 21:54:17
28.。。。。。。。。这个是 2^48 次方
因为我们的相位累加器，是 48 位的。后面的 100.。。。是 100MHz，这个是我们得采样率。
王紫豪-XiaomaGee(15959622) 21:55:22
关于 dds 输出频率的计算，它符合如下公式：

$$f_{out} = \frac{m \times f_{sample}}{2^n}$$

上面这段程序，也是这个公式的解析。公式是我博客教程中的截图。
王紫豪-XiaomaGee(15959622) 21:56:49
fout 代表输出频率， m 是频率子,fsample 是采样时钟,我们这里是 100MHz，2n 次方，这个 n 是相位累加器的位数；这里是 48bit

王紫豪-XiaomaGee(15959622)　21:57:49
这个公示中还隐藏着这一层含义，就是我们任意波发生器输出频率分辨率可达　0.35uHz，也就是　0.35　微 Hz
王紫豪-XiaomaGee(15959622)　21:58:59

```
fpga_ctl.b.ctl1 = 0;
fpga_ctl.b.ctl2 = 0;
fpga_ctl.b.ctl3 = 0;
fpga.ctl();

FPGA_WR2 = (fword >> 32) & 0xffff;
```

这样的程序，是 stm32　对 fpga 控制的代码，他通过　16bit 总线和地址译码的方式，来写入　fpga
16bit 总线，所以　48 位需要写入三次。
王紫豪-XiaomaGee(15959622)　22:00:05
下面我们来讲解波形数据的产生代码，请打开　waveform.c
王紫豪-XiaomaGee(15959622)　22:01:53
waveform.c 里面仅仅有一个　set 函数，它包含两个输入参数，分别为　波形和第二参数。
第一个参数波形，就是我们设定波形的类型，正弦波、方波、三角波等等。
第二个参数，是诸如方波的占空比，三角波的对称度等等参数。
王紫豪-XiaomaGee(15959622)　22:03:04

```
case SINE:
  for (i = 0; i < 4096; i++) {
    temp = i;
    temp /= 4096;
    temp *= 2;
    temp *= 3.1415926;
    temp = sin(temp);
    temp += 1;
    temp /= 2.0;
    temp *= 255;

    FPGA_WR2 = temp;
  }
  break;
```

这一小段程序，就是生成一个正弦波（4k 采样点），然后写入到 fpga　ram 内的过程。写入后，dds 内核负责直接输出。
王紫豪-XiaomaGee(15959622)　22:04:07
其实，他主要使用了　math.h 库里面的　sin 函数来生成。这个方法在　51　单片机是行不通的，因为速度慢。但是我们的 stm32 性能强劲，他可以几乎瞬间输出　四千（4000）　的计算值。
王紫豪-XiaomaGee(15959622)　22:05:13
同样的方法，我们实现了十几种波形的产生，当然，如果谁有兴趣，也可以添加自己自定义的波形；这就是所谓的“任意波形发生器”的优势，需要什么波形，直接写入就可以了。

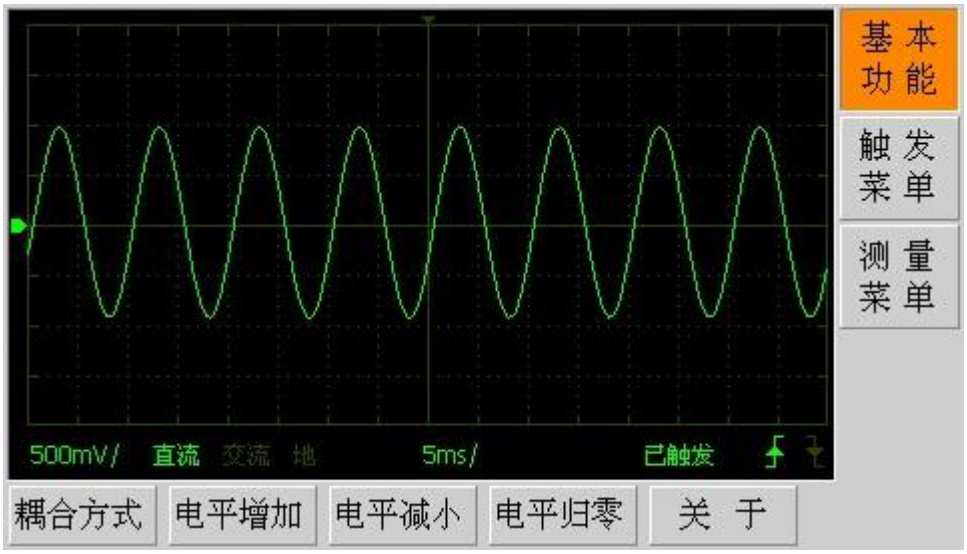
王紫豪-XiaomaGee(15959622)　　22:06:25
例如　sinc　函数，数据生成函数如下：

```
case SINC:
  for (i = -2048; i < 2048; i++) {
    temp = 20. * PI;
    temp /= 4096.;

    temp *= i;
    temp1 = temp;
    temp = sin(temp);
    temp /= temp1;

    temp += 1;
    temp *= 128;
    if (i == 0)
      temp = 255;
    temp2 = temp;

    FPGA_WR2 = temp2;
  }
```

通过自带的示波器，我们可以测量到自己产生的波形

王紫豪-XiaomaGee(15959622)　　22:07:28
只是被菜单盖住了，哈哈
王紫豪-XiaomaGee(15959622)　　22:09:18
这是正弦波，通过自带的示波器查看

完全自产自销：



QQ 截图后直接会进行 jpg 的压缩，所以图片失真，原始图片比大家看到的这个清晰的多。

王紫豪-XiaomaGee(15959622) 22:10:42

波形生成过程，就讲解到这里，不清楚的网友下课了再问。

王紫豪-XiaomaGee(15959622) 22:11:17

下面我们看一下幅度控制代码部分

再 amplitude.c 里，大家打开。

```
const AMPLITUDE_T amplitude = {
    .set = set
};
```

同样，也是仅仅有一个 set 函数

王紫豪-XiaomaGee(15959622) 22:12:34

《iBoard infinity》固件包编程风格都是一样的，所以大家看懂一个，其他都懂了。

```
static int set(double amp)
{

  if (amp > 10. || amp < 0.)
    return -1;


  if(amp > 3.16){
    amp *= 1.;
    ATT_0DB;
  }else if(amp > 1.){
    amp *= 3.16;
    ATT_10DB;
  }else if(amp > 0.316){
    amp *= 10.;
    ATT_20DB;
  }else if(amp > 0.1){
    amp *= 31.6;
    ATT_30DB;
  }else if(amp > 0.0316){
    amp *= 100.;
    ATT_40DB;
  }else if(amp > 0.01){
    amp *= 316.;
    ATT_50DB;
  }else {
    amp *= 1000.;
    ATT_60DB;
  }

  amp /= 11.23;
  amp *= VREF;

  dac.arb_amp = amp;
  dac.set();
  return 0;
```

王紫豪-XiaomaGee(15959622) 22:13:35
这是整个函数体。。。这里加入了输出幅度的自动衰减判断法；提高了小信号的信噪比；
我简单的说下这个方法：
假如我们输出 10V，则就是原始输出；如果我们输出 1V，则是前端输出 10V，然后衰减 10 倍（20dB）得来的。
王紫豪-XiaomaGee(15959622) 22:15:17
当然，这个需要配合硬件上的衰减网络电路。。

王紫豪-XiaomaGee(15959622) 22:15:37
我的代码风格是c语言创始人 k&r的风格
寒雨(251539157) 22:15:46
我说你的代码风格很好
王紫豪-XiaomaGee(15959622) 22:15:54
也就是linux内核代码的风格。。
寒雨(251539157) 22:15:57
就是多用tab
王紫豪-XiaomaGee(15959622) 22:15:57
谢谢，哈哈

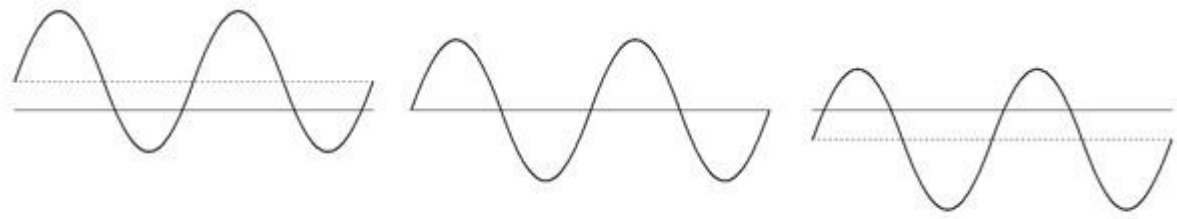
王紫豪-XiaomaGee(15959622) 22:16:54
这里我说一下任意波发生器的幅度校准，我们只需要改变：

```
amp /= 11.23;
amp *= VREF;
```

这里的 11.23值，就可以做到校准的目的。
所以，收到货，输出有微小偏差的网友，可以通过此方法校准。
半闲(602227008) 22:18:01
11.23 是怎么来的？
王紫豪-XiaomaGee(15959622) 22:18:08
是一个系数
跟电路有关
其实，这个 dac的值，是通过 stm32不断的刷出来的。这段说明，可以看下我的博客有篇文章,通过 stm32的一路 dac,扩展了7路输出。
http://www.cnblogs.com/xiaomagee/archive/2012/03/18/2405248.html
王紫豪-XiaomaGee(15959622) 22:20:27
同样的方法，我们可以通过直流电平去调节波形输出的直流偏置

```
//---------------- variabl
const OFFSET_T offset = {
  .set = set
};
```

王紫豪-XiaomaGee(15959622) 22:21:50



图二（1）偏置为正电压　图二（2）偏置为零电压　图二（3）偏置为负电压

这个就是直流偏置的含义。

```
static int
set(double offs)
{
  if (offs > 5. || offs < -5.) return -1;

  offs /= 5.;

  offs += 1.24;

  dac.arb_offset = offs;
  dac.set();

  return 0;
}
```

实现代码如上。。。。
王紫豪-XiaomaGee(15959622) 22:24:02
至此，一个简单的波形发生器 c 代码我就说完了，当然，细节性的问题还需要大家自己学习。

c 语言是一个需要长期学习、训练的过程，只要大家多看，多想和多练，就会达到炉火纯青、应用自如境界。
王紫豪-XiaomaGee(15959622) 22:24:56
代码还需要自己看，我也就是讲讲构架。
木木糖(896539497) 22:25:10

 马哥辛苦了

IC 爱好者(446136066) 22:25:17

王紫豪-XiaomaGee(15959622) 22:25:58
我的代码，都很标准，风格也同意；新手可以参考

小马哥和网友交流环节；

 灰太狼(28254042) 22:28:01

arb.c 中跳入 function.c 是在何处
王紫豪-XiaomaGee(15959622) 22:28:17
执行了 function.main()
淡泊名利，宁静致远！(591765165) 22:30:52

```
if (first_run == 1) {
for (j = 0; j < 100000; j++) ;
for (i = 0; i <= 100; i++) {
pwm.initialize(i);
for (j = 0; j < 20000; j++) ;
first_run = 0;
}
```

实现什么功能？ 呵呵

王紫豪-XiaomaGee(15959622) 22:31:15
实现背光逐渐变亮的效果。。
淡泊名利，宁静致远！(591765165) 22:31:55
哦，这个是调节背光灯的，呵呵，我开机时 没感觉到，
王紫豪-XiaomaGee(15959622) 22:32:11
这个效果很好
淡泊名利，宁静致远！(591765165) 22:33:40
static int
_waveform(void)，函数前面加 static，的作用？
心理委员^_^(772880135) 22:33:58
静态。

淡泊名利，宁静致远！(591765165) 22:34:12
嗯 定义成静态 的好处？
王紫豪-XiaomaGee(15959622) 22:34:16
静态，仅本文件有效；不冲突
戴布魅快(535526369) 22:34:39
作用域
王紫豪-XiaomaGee(15959622) 22:34:46
对
淡泊名利，宁静致远！(591765165) 22:34:54
函数里面的静态变量 就是本函数里有效，对不？
呵呵
戴布魅快(535526369) 22:35:09
不。
心理委员^_^(772880135) 22:35:16
对
戴布魅快(535526369) 22:35:22
是周期
心理委员^_^(772880135) 22:35:35
？
淡泊名利，宁静致远！(591765165) 22:35:42
但是 函数里的 int 变量 好像也是本函数有效？
戴布魅快(535526369) 22:35:49
运行周期