把 HP 39gs 计算器变成 ARM9 开发板！（初探篇）

◇张文挺

HP 39gs 是何方神圣？

HP 39gs 是惠普公司推出的一款面向学生和工程师的图形计算器。什么是图形计算器？它是一种既能计算又能作图的新型数学工具，它具备数学运算系统、几何操作系统、数据分析系统等，可以直观地绘制各种图形，并进行动态演示、跟踪轨迹。它还包括了几何学、统计、金融及其他许多课堂常用的功能。通过连接各种传感器，还可进行科学实验。由于拥有大闪存，可以保存多个程序，通过联机电缆可将机内的操作系统升级。通过图形计算器可以实现如解方程、编程、作图等普通计算器无法完成的功能。

说白了，它就是一种很高级的计算器，可以自己在上面写程序运行。我们这次使用的计算器是 HP 39gs。

HP 39gs 的 CPU 用的虽然是 ARM920T 的 MPU，内置 MMU，但是由于存储空间太小，基本只能当成 MCU 来使用。不过这么设计也不是没有它的原因，39gs 和其他更高阶的型号（如 48gII 和 49g+）是 HP 在 21 世纪初推出的 HP 39G、48S/GX 系列的升级款机型。HP 39G 和 48S/GX 使用的是 HP 自有的 Saturn 处理器，专门为计算器设计，64bit/20bit 混合设计，但是使用 4bit 指令集。这种处理器最初被使用在一款手持 BASIC 编程计算器 HP-71B 上，后来 HP 为它专门打造了 RPL 语言，用于 Saturn 处理器的开发。HP 3 系列和 4 系列的系统都是用 SysRPL 语言开发的，在计算器内，用户可以使用 User RPL 语言进行第三方程序的开发。这套设计相当经典，其高效性吸引了很多人成为惠普计算器的粉丝。然而进入 21 世纪，HP 原来的计算器工作速度实在是太慢了，HP 可以选择用更好的制程重新打造 Saturn 处理器，但是由于成本原因，HP 决定使用高主频的 ARM 处理器来模拟 Saturn 处理器，虽然会损失运行效率，但是毫无疑问会比原来的 Saturn 快得多。因为是模拟器，所以自然不需要太多的资源，反而因为 20 世纪的计算器都是不带可擦写存储器的，数据都会被保存在 SRAM 当中，由一颗纽扣电池来保留数据。现在运行模拟器，为了保证内存的数据不会丢失，HP 选择继续使用 SRAM 作为主内存而不是便宜得多的 SDRAM。

如上面所说，HP 39gs 是支持直接编程的，但是程序只能在机器内置的 Saturn 模拟器中运行，这当然不是我们想要的，所以，肯定要对计算器进行一些改造，首先得“废”了它的模拟器。

Tips： HP 39gs的简要硬件配置情况

CPU： 三星S3C2410A，ARM920T最大208MHz主频

RAM： 256KB SRAM

ROM： SST39VF800，1MB NOR FLASH

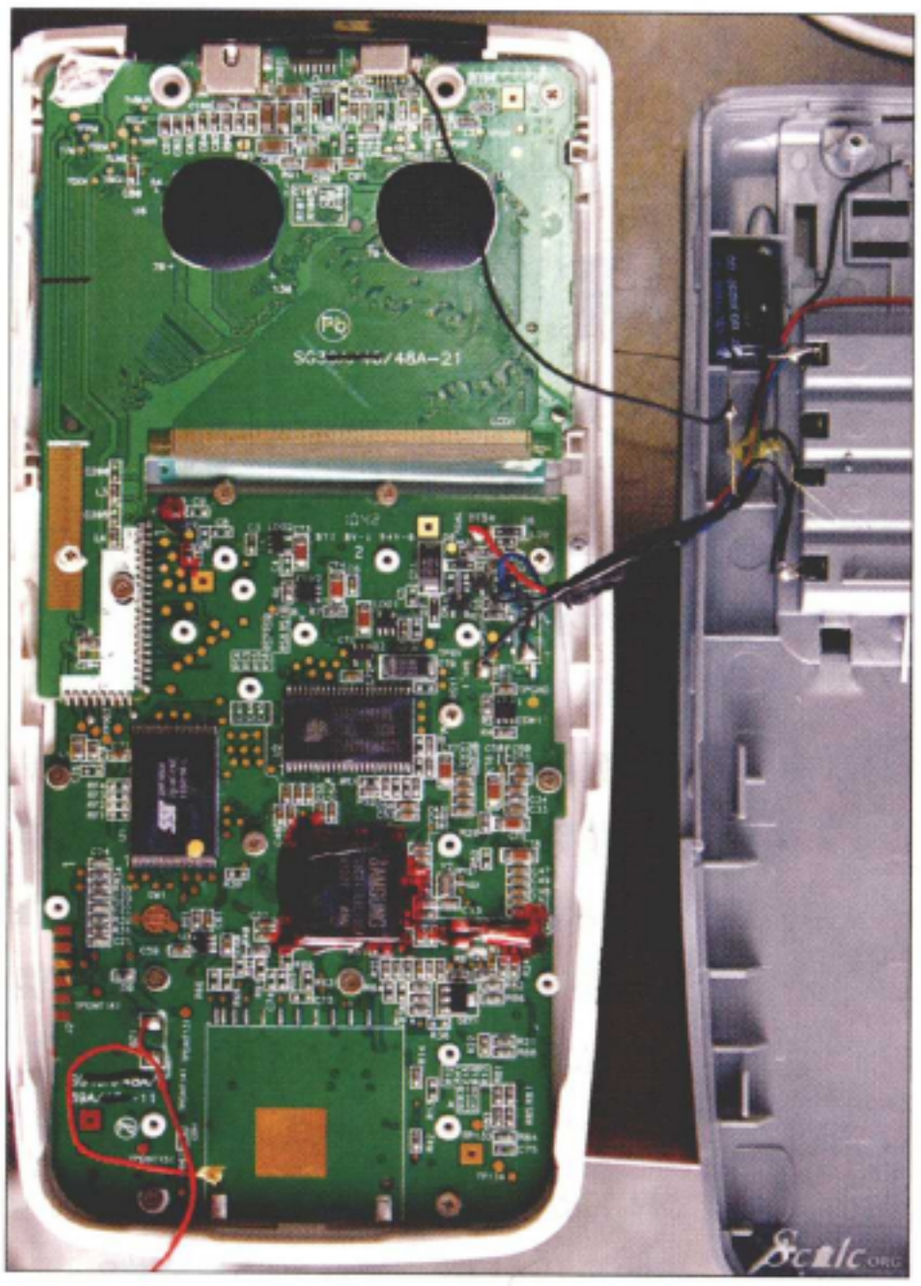
屏幕： 132像素×64像素 16级灰度FSTN液晶

拆解！

单有个计算器肯定是没法开发的，一般来说，我们得找出 JTAG 接口，这样才能把自己的程序下载进去。

要拆机首先得把屏幕前面板拆下来，然后拧下两枚螺丝。整个机子是用卡扣固定的，会很紧，请小心拆开。然后注意蜂鸣器的连线和电池仓的连线，把屏蔽纸拿下来就能看到电路板了，图 1 所示是电路板的照片。

可以看见，HP 在电路板上留了超多的测试点，不过首先引起我注意的是左边的那一排引脚，会不会是 JTAG 呢？笔者经过了各种的尝试，已经可以告诉大家，它就是 JTAG！现在公布它的线序，从上到下依次是：TCK、TDI、TDO、TMS、GND、GND，这样就可以用 Jlink 连上它了！记得把旁边的 RESET 也连接上。



■ 图 1 HP 39gs 内部电路板

逆向工程

现在我们只是打通了和 CPU 的桥梁，关于其他的如屏幕、键盘、蜂鸣器之类的我们一概不知道。没关系，我们可以从原厂系统中提取这些信息。

系统开机分为几个阶段：模拟器程序初步初始化硬件、加载 ROM、模拟的 ROM 的程序重新初始化硬件。模拟器在初始化完成后会开启 MMU，对内存进行重映射，干扰我们的分析，所以要赶在加载 ROM 前停止 CPU 的运行。首先打开计算器，让它正常开机，打开 J-Link Commander 软件，此时应该可以看到如图 2 所示界面。

■ 图 2 J-Link Commander 软件界面

注意，图 2 中有一句“Found ARM with core Id 0x0032409D (ARM9)”，此时计算器已经正确被 J-Link 识别。现在输入 r，按回车键进行重启，计算器会重启并且停止运行，输入 g，回车，让 CPU 开始执行指令，大约 1s 后输入 h，回车，让计算器停止执行指令。现在可以开始分析了。先分析什么呢？我们从简单的开始，先研究研究蜂鸣器吧。还好，这款计算器在开机的时候会“哔”的叫一声，我们得让它在叫的时候将 CPU 停下来，这样才能分析。于是按 r 键重启，等叫的瞬间按 h，回车，停止 CPU，这样会一直叫下去，如果没成功就再试一次吧。我试了 10 多次才成功停在那个地方。

根据以前玩单片机的经验，蜂鸣器一般是通过 PWM 控制的，那么我们就来研究一下。首先请准备一份文档——S3C2410A 用户手册，这个在网上很容易下载到。然后打开 J-Mem 工具，也是 J-Link 自带的工具，用于查看内存。

既然是研究 PWM，我们就找到 S3C2410 手册中 PWM 的部分，注意到一个关键的寄存器，用来控制各个 PWM 的开关，如图 3 所示。

S3C2410A PWM TIMER

TIMER CONTROL (TCON) REGISTER

Register	Address	R/W	Description	Reset Value
TCON	0x51000008	R/W	Timer control register	0x00000000

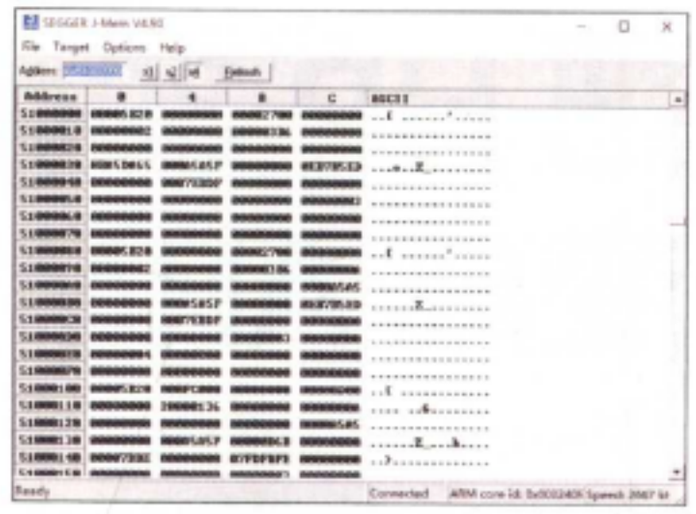
■ 图 3 控制 PWM 开关的寄存器

我们直接在 J-Mem 上方输入 PWM 定时器的基地址 0x51000000，应该能看到如图 4 所示的场景。

如果蜂鸣器在叫，这些寄存器应该是这样的：

TCFG0=0x00002C16

TCFG1=0x00033333



■ 图 4 在 J-Mem 上方输入 PWM 定时器的基地址（注：截图时蜂鸣器没有响）

TCON=0x00498909

TCNTB0=0x0000FFFF

TCMPB0=0x00003FF6

TCNTB1=0x000001FF

TCMPB1=0x000000FF

TCNTB2=0x00000013

TCMPB2=0x00000009

TCNTB3=0x00000014

TCMPB3=0x0000000A

TCNTB4=0x00000209

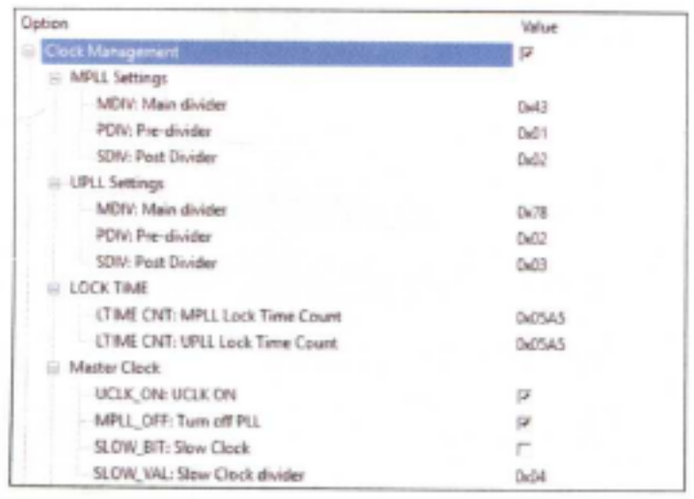
TCMPB4=0x00000207

也就是说，4 个定时器全部打开了，通过分别关闭，我们可以知道，是 PWM2 连接到了蜂鸣器，剩下的，未知。别忘了，还有 GPIO，GPIO 必须要打开，PWM2 的输出才响！

接下来，我们可以写个小程序，实践一下我们刚刚推出的那么一点点成果。

实战

我们这次使用的编译器是 Keil MDK-ARM，新建工程的方法不再赘述，可以用我建好的，也可以自己建。需要注意的是



■ 图 5 时钟配置

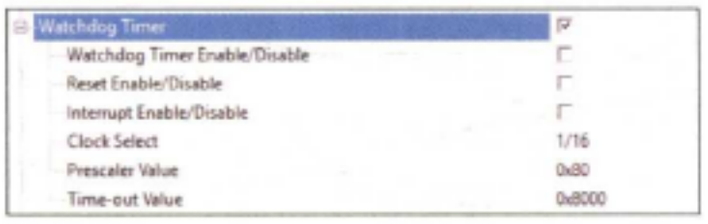
启动文件的配置，注意像我这样配置。

首先是时钟，主频为 75MHz，USB 为 48MHz，如图 5 所示。

在 CLOCK GENERATION 里面把要用的打开，比如 GPIO、PWM TIMER，如图 6 所示。全打开也无所谓。

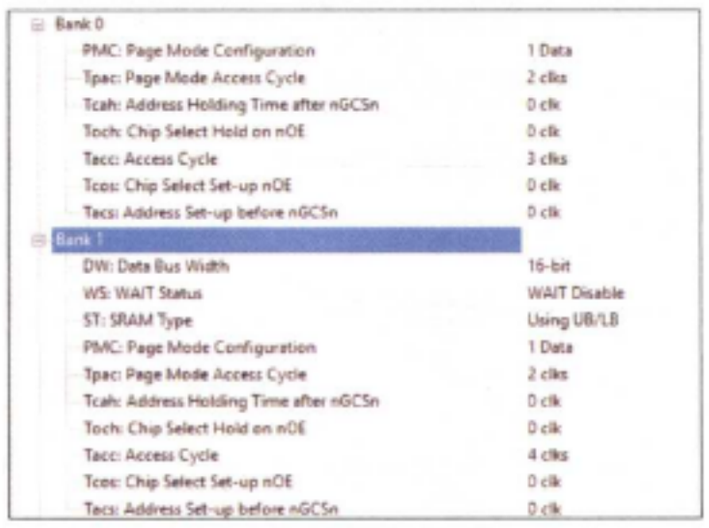
■ 图 6 CLOCK GENERATION 界面

记得把 WATCHDOG TIMER 的配置也勾上，如图 7 所示，不要打开，不然 WDT 默认是打开的，1s 后就会自动重启。



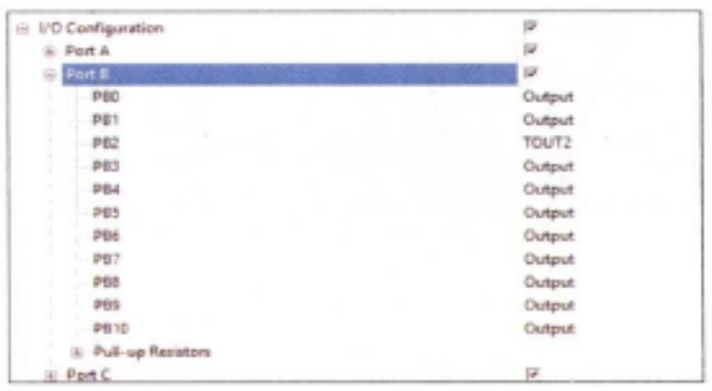
■ 图 7 WATCHDOG TIMER 的配置

内存控制器部分，打开前两个 Bank（0 是 NOR，1 是 SRAM），如图 8 所示。



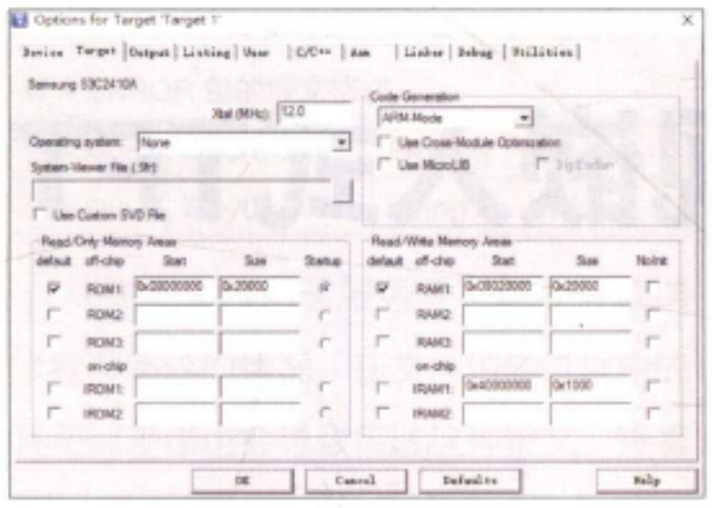
■ 图 8 内存控制器配置

I/O 部分注意打开 GPB2，这是连接蜂鸣器的端口，如图 9 所示。



■ 图 9 I/O 部分的配置

之后，工程部分也要配置一下。我们先不去动那个 Flash，让程序在 RAM 中运行。像图 10 所示那样填写就可以了。其实就是把 RAM 划出一块来，当成 ROM 用。



■ 图 10 工程部分的配置

OK，开始写代码。关于蜂鸣器，我写了两个函数，分别是开始和结束。

```
void Buzzer_Start(unsigned long
freq)
{
        rGPBCON &= ~(3<<4);
        rGPBCON |= (2<<4);
        rTCFG0  &= ~0xff00;
        rTCFG0  |= 0x0f00;
        rTCFG1  &= ~0x0f00;
        rTCFG1  |= 0x0200;
        rTCNTB2 = (PCLK>>7)/freq;
        rTCMPB2 = rTCNTB2 >> 1;
        rTCON &= ~0xF000;
        rTCON |=  0xB000;
        rTCON &= ~0x2000;
}
void Buzzer_Stop()
{
        rGPBCON &= ~(3<<4);
        rGPBCON |= (1<<4);
        rGPBDAT &= ~(1<<2);
        rTCON   &= ~0xF000;
}
```

在主程序里面写上 Buzz_Start(2000)，编译后按 DEBUG 键，看看是不是有声音了？调节这个参数就可以调节频率。不如来试试放首歌吧！（感觉就像刚学 51 单片机的时候一样。）

播放程序直接使用 Arduino 的例程。主程序稍微修改下，以配合我们的函数。

```
        int thisNote =0;
        int noteDuration;
        for (thisNote = 0; thisNote
< 372; thisNote++)
{ noteDuration = 2500/noteDurations
[thisNote];
        Buzzer_
Start(melody[thisNote]);
        Delay(noteDuration);
        Buzzer_Stop();
}
```

就这样，放在 main 当中，然后把歌曲写进去，我就放一小段吧，是 BILIBILI 的入站歌。

```
int melody[] = {
NOTE_E5, NOTE_B5, NOTE_E5, NOTE_B5,
NOTE_A5, NOTE_GS5, NOTE_E5, NOTE_
E5, NOTE_B5, NOTE_E5, NOTE_B5,
NOTE_A5, NOTE_B5, NOTE_CS6, NOTE_
E5, NOTE_B5, NOTE_E5, NOTE_B5,
NOTE_A5, NOTE_GS5, NOTE_E5,
NOTE_GS5, NOTE_A5, NOTE_GS5, NOTE_
FS5, NOTE_FS5, NOTE_GS5, NOTE_FS5,
NOTE_E5, NOTE_FS5, NOTE_E5,
NOTE_B5, NOTE_E5, NOTE_B5, NOTE_A5,
NOTE_GS5, NOTE_E5, NOTE_E5, NOTE_
B5, NOTE_E5, NOTE_B5};
int noteDurations[] = {
16,8,16,8,8,8,8,16,8,16,8,
8,8,8,16,8,16,8,8,8,8,
4,16,16,8,16,32,32,16,16,16,
8,16,8,8,8,8,16,8,16,8,
};
```

上面一个是音阶表，下面一个是音长表，两者是一一对应的，这样基本就可以把简谱翻译进去放啦。点击 DEBUG，就可以听到音乐声了。

下一期，我们来讲一讲 LCD 的驱动和怎么把自己写的程序固化到计算器里面吧。