miniDSP Codec 初始化方法及示例代码

王凡 Ryan Wang (Fan) South China and Shenzhen OEM Team

摘 要

Texas instruments 推出的超低功耗 miniDSP 音频 Codec 集成了 miniDSP 内核，可在耗电极低的工作状态下为电池供电的便携式产品提供高性能的语音及音乐处理能力。本文详细介绍了如何初始化 miniDSP Codec 并提供了基于 MCU 控制器的参考代码。

目 录

图 表

Preliminary

1 miniDSP Codec 寄存器架构

miniDSP Codec 内部的寄存器分为两大类: 常规配置寄存器和 miniDSP 内存寄存器。常规配置寄存器是用来控制 Codec 时钟、电源、路径等常规设置的普通寄存器。miniDSP 内存寄存器是 miniDSP 内存映射到 I2C/SPI 控制端口的寄存器地址，主控制器可以通过该地址进行初始化及控制。

miniDSP 的内存分为三种类别：

- 指令内存（Instruction RAM）用于存储 miniDSP 的运行指令及程序。
- 数据内存（Data RAM）用于暂存 miniDSP 运行时的运算结果等临时数据。
- 系数内存（Coefficient RAM）用来存储 miniDSP 音效、增益等控件的参数设置。

指令内存和系数内存均可通过映射的 I2C 或 SPI 地址来进行读写。miniDSP Codec 完整的初始化过程包括对常规配置寄存器的配置和对 miniDSP 内存加载代码两个部分。

1.1 miniDSP Codec 初始化方式介绍

miniDSP Codec 具有两种初始化方式：

- 预置处理模式：

为了简化 miniDSP Codec 的配置，miniDSP Codec 内部存储了若干组预置处理模式（Processing Block，简称 PRB）的代码，不同的预置处理模式提供了不同的音效处理能力及功耗特性。用户无需对 miniDSP 进行编程，只要按需求选择一个预置处理模式即可完成配置。

- miniDSP **编程模式：**

miniDSP 编程模式下，用户需要按照音频处理需求开发 miniDSP 的程序。初始化时将程序通过 I2C/SPI 加载到 miniDSP 内存中运行。相比预置处理模式，miniDSP 编程模式下 Codec 的功能更强，使用更灵活性。

Preliminary

2 预置处理模式的初始化过程

miniDSP codec 提供了多个预置处理模式供用户选择，不同的预置处理模式支持不同的采样率、音效处理模块、功耗等级等参数。预置模式的详细信息请参考相应器件的手册。

选择了预置处理模式工作的 miniDSP codec 初始化过程和传统不带 miniDSP 的 codec 类似，流程图 1 给出了推荐的初始化流程供参考：

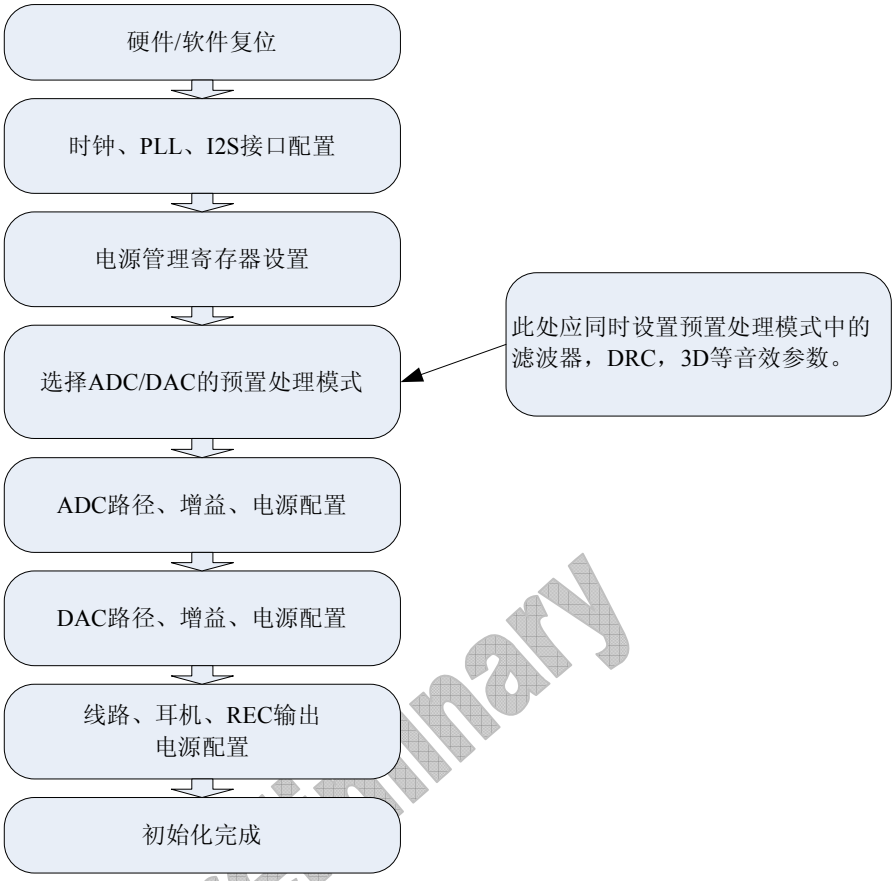

图 1. 预置处理模式的初始化流程图

Preliminary

3 miniDSP 编程模式的初始化过程

为了简化 miniDSP 的软件开发工作，TI 推出了独有的图形化编程开发环境 PurePath™ Studio (Portable Audio) Graphic Development Environment(简称,GDE)。GDE 支持的图形化界面可以完成 miniDSP 的音频处理程序的开发，并生成初始化代码供程序使用。关于如何使用 GDE 生成初始化代码，请参考文档：使用 miniDSP Codec 提升智能手机的音频效能（ZHCA113）。

3.1 初始化头文件结构

初始化头文件结构如图 2 所示，主要分为接口段和代码段两部分。接口段主要包括每个音量、开关、音效控件的名称信息及地址信息，用来提供给驱动程序一个接口来控制它们。代码段包括三个子段落：常规配置寄存器代码段、miniDSP-A 程序代码段和 miniDSP-D 程序代码段。驱动程序需要引用代码段内的数组来完成常规配置寄存器及 miniDSP 的程序写入。

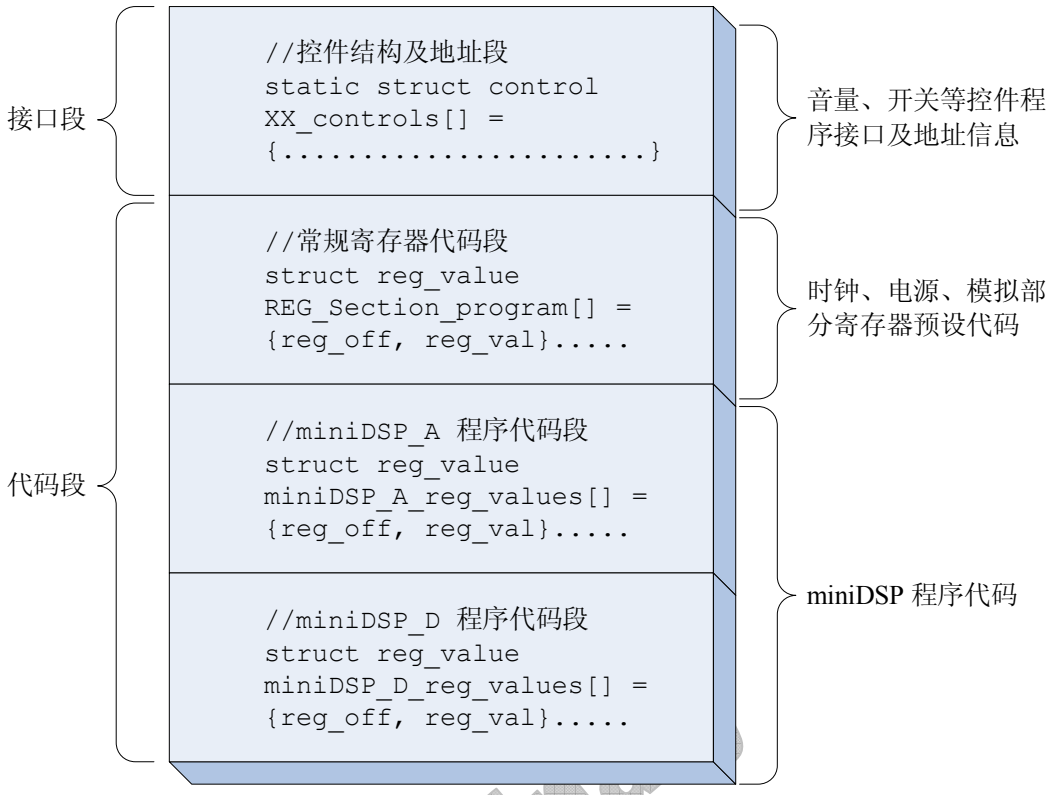

图 2. 驱动头文件结构示例

需要注意的是常规配置寄存器代码段内的参数通常是 GDE 按照器件 EVM 板产生的默认参数，该参数并不一定适用于实际的用户系统。例如用户系统使用了与 EVM 板不同的 MCLK 时钟频率，则用户必须手动更改常规配置寄存器代码段内的 PLL 参数来满足实际需求。用户可通过修改 GDE 的器件 Frameworks 中默认的 System Settings Code 来产生满足目标应用的头文件。

3.2 代码下载规则

初始化头文件数组内元素的结构为：

{寄存器地址，寄存器值}

用户需按照以下规则将数组内的值通过 I2C/SPI 加载到 miniDSP Codec 中：

- 常规寄存器代码段：

该段寄存器内容均为常规配置，例如电源、时钟、路径配置等。主控按照先后顺序将寄存器写入。需要注意的是，GDE 生成的常规寄存器代码段包括部分控制命令，控制器需要识别后进行相应操作，以下为控制命令的详细解释：

  – 若寄存器地址=254：延迟命令，延迟时间为寄存器值（十进制），单位为毫秒。
  – 若寄存器地址=255，寄存器值=0：跳转到 miniDSP-A 代码段，进行 miniDSP-A 的代码下载。

– 若寄存器地址=255，寄存器值=1：跳转到 miniDSP-D 代码段，进行 miniDSP-D 的代码下载。

流程图 3 给出了详细的常规配置寄存器数组的解析下载过程，例程请参考附录 A。

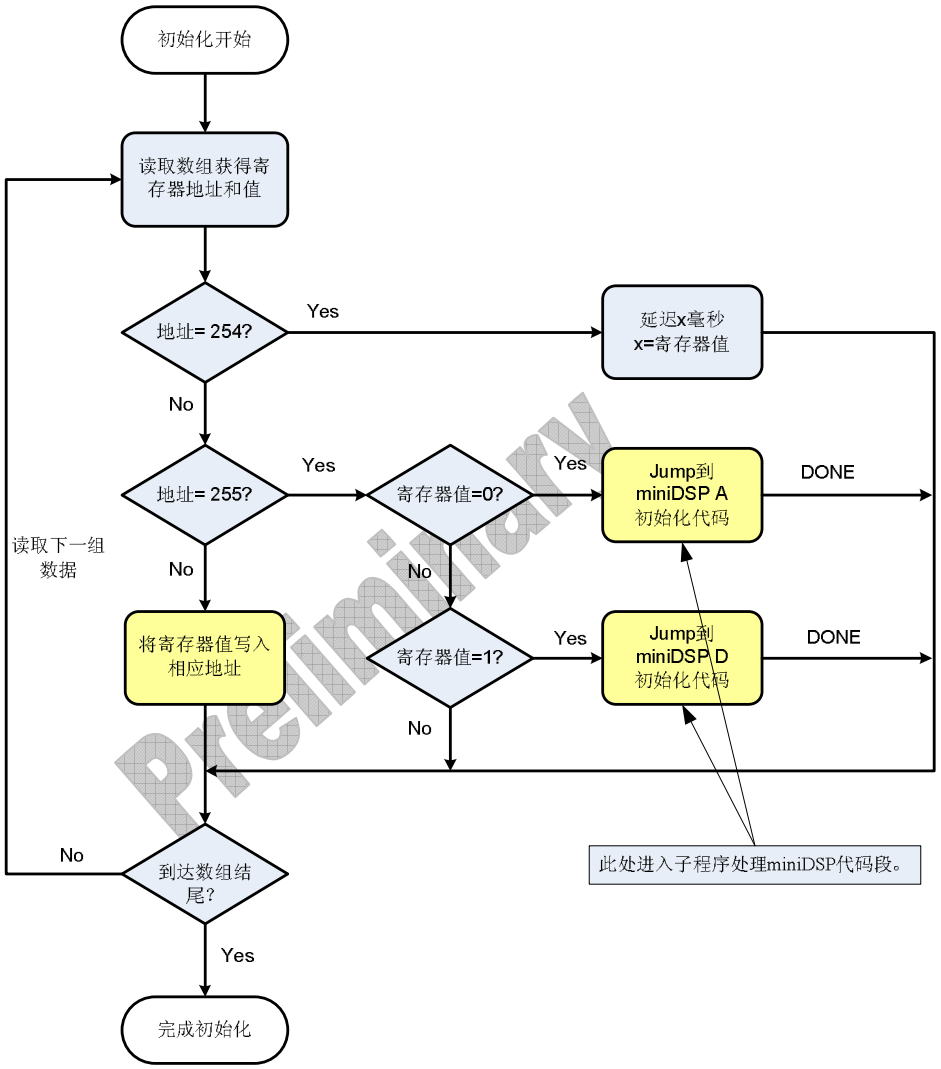

图 3. 常规寄存器代码段下载流程

- miniDSP-A 和 miniDSP-D 代码段：

miniDSP-A 和 miniDSP-D 的数组数据按页排列，每页中最多有 120 个连续数据（Reg 8 ~ Reg127）。miniDSP Codec 支持 I2C/SPI 连续写入模式，当写入第一个数据后，不释放控制总线，继续写入后续数据，器件内部寄存器地址会自动递增，最大可完成一个完整页面的写入。所以为了节省 I2C/SPI 的写入时间，该部分代码建议使用连续写入方式进行初始化。流程图 4 给出了连续写入模式的下载时序。例程请参考附录 A。

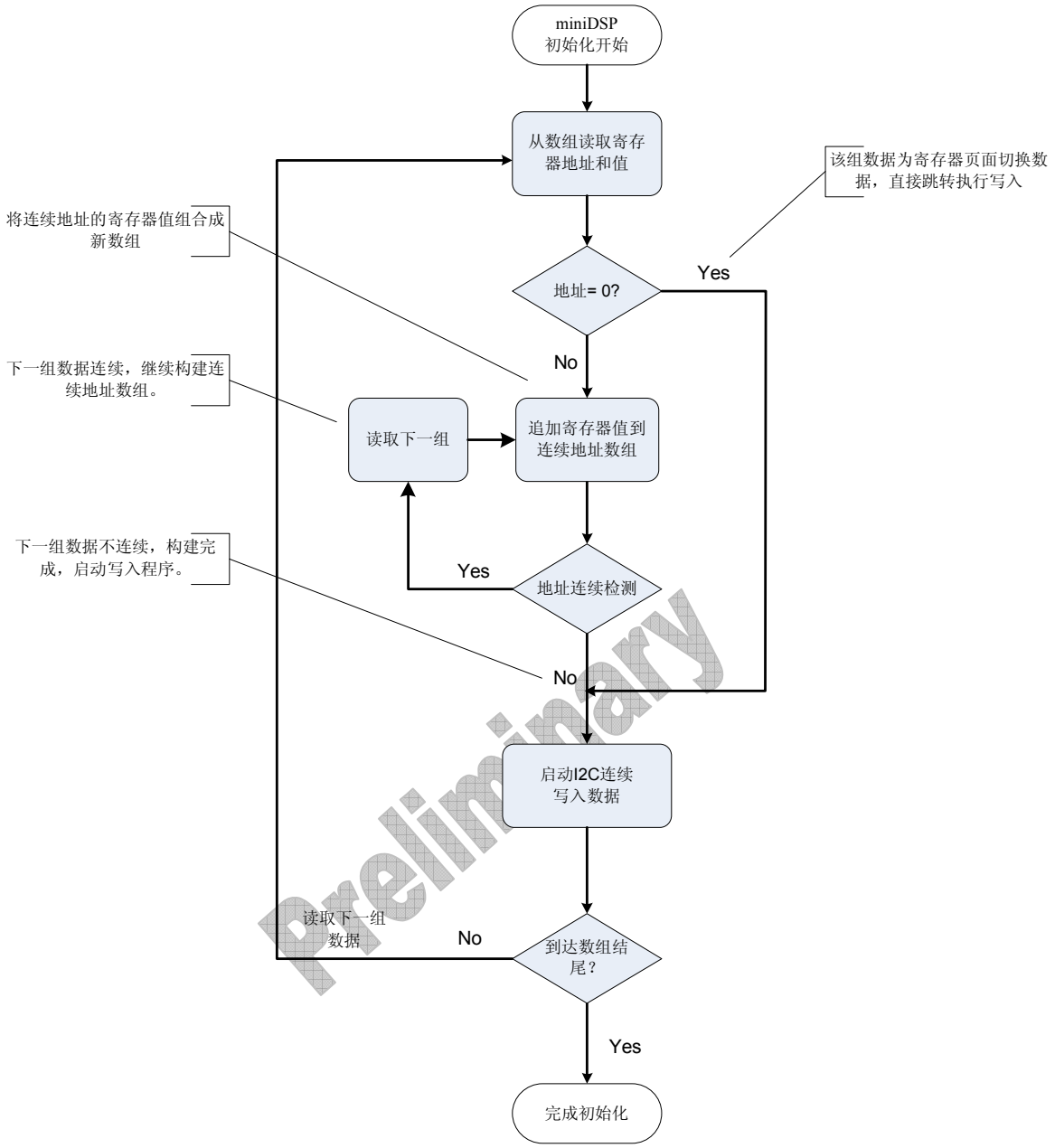

图 4. miniDSP 代码段下载流程

4 存储空间需求

由于 miniDSP Codec 寄存器数据量较大，某些应用中使用的低 Flash 容量的单片机可能出现存储空间不足的情况。所以在设计之初就需要初步计算 miniDSP 所需代码量。GDE 生成的头文件中包含有 miniDSP-A 和 D 的数据量信息。如下所示：

```
reg_value miniDSP_A_reg_values[] = {......};
#define miniDSP_A_reg_values_COEFF_START   0
#define miniDSP_A_reg_values_COEFF_SIZE    1816
#define miniDSP_A_reg_values_INST_START    1816
#define miniDSP_A_reg_values_INST_SIZE     3118

reg_value miniDSP_D_reg_values[] = {......};
#define miniDSP_D_reg_values_COEFF_START   0
#define miniDSP_D_reg_values_COEFF_SIZE    1590
#define miniDSP_D_reg_values_INST_START    1590
#define miniDSP_D_reg_values_INST_SIZE     3247
```

该例中给出了 miniDSP-A 和 D 的 COEFF_SIZE 和 INST_SIZE， 总内存占用计算方式为：

（miniDSP_A_reg_value_COEFF_SIZE
+ miniDSP_A_reg_values_INST_SIZE
+ miniDSP_D_reg_values_COEFF_SIZE
*+ miniDSP_D_reg_values_INST_SIZE) * 2*

该例的结果约为 7.142KB，则需要 MCU 有至少 8kB 的 Flash 空间储存完整的头文件数组内容。

实际应用中，miniDSP 也有可能需要多种工作模式，例如用户选择 SRS 模式，或者 Dolby 模式。这种多工作模式的方式需要注意 MCU 需要同时存储所有工作模式的头文件，对 MCU 的 Flash 容量要求较高。

用户有两种选择来解决 Flash 内存不足的问题：

- 当头文件容量超过 MCU 存储空间较多时，可以采用外置的 EEPROM 来存储头文件的二进制数据。MCU 只需要负责启动时将头文件从 EEPROM 调出来配置 Codec。该种模式可以低成本提供较大的数据空间。
- 若头文件容量超出 MCU 存储空间不多时，可以简化头文件数组来缩小大小。MiniDSP-A 和 D 的数组是分段连续的，可以将连续的地址位数据省略，只保留数据位的值来简化头文件。这种方法可以简化大约 40%的数据量。

Preliminary

5 总结

本文详细介绍了 miniDSP Codec 的内部架构及初始化方法，并以 AIC325x 为例提供了基于 MCU 的参考例程。用户可参考本文档快速开发 miniDSP Codec 的初始化程序。

6 参考文献

1. TLV320AIC3254, Ultra Low Power Stereo Audio Codec With Embedded miniDSP-Data sheet (SLAS549)
2. Design and Configuration Guide for the TLV320AIC3204 & TLV320AIC3254 Audio Codec (SLAA404C)
3. Coefficient RAM Access Mechanisms (SLAA425A)
4. 使用 miniDSP Codec 提升智能手机的音频效能（ZHCA113）

Appendix A. 参考代码

```
/*
 *---------------------------------------------------------------------------
 * Function: process_flow_download
 * Purpose:  download the head file which generated by GDE.
 *---------------------------------------------------------------------------
 */

void process_flow_download (reg_value *REG_Section, int program_size)
{
    static int reg_index = 0;
    for ( ; reg_index < program_size; reg_index++){
         if (REG_Section[reg_index].reg_off == 254){
             Delay_1ms(REG_Section[reg_index].reg_val);
             continue;
         }
         if (REG_Section[reg_index].reg_off == 255){

             if (REG_Section[reg_index].reg_val == 0){
    // Program_miniDSP_A;
    minidsp_burst_transfer (miniDSP_A_reg_values, miniDSP_A_reg_values_COEFF_SIZE + miniDSP_A_reg_values_INST_SIZE);
             }

             if (REG_Section[reg_index].reg_val == 1){
    // Program_miniDSP_D;
    minidsp_burst_transfer (miniDSP_D_reg_values, miniDSP_D_reg_values_COEFF_SIZE + miniDSP_D_reg_values_INST_SIZE);
             }
             continue;
         }
             aic325x_write(REG_Section[reg_index].reg_off, REG_Section[reg_index].reg_val);
     }
}


/*
 *---------------------------------------------------------------------------
 * Function: minidsp_burst_transfer
 * Purpose:  miniDSP data burst transfer to increase download speed
 *---------------------------------------------------------------------------
 */

void minidsp_burst_transfer(const reg_value *program_ptr, int program_size)
{
    minidsp_parser_data parse_data;
    /* point the current location to start of program array */
    parse_data.current_loc = 0;
    parse_data.page_num = 0;
    parse_data.reg_num = 0;

    do {
        /* Prepare burst data */
        minidsp_get_burst(program_ptr, program_size, &parse_data);
        /* Send burst data */
        aic325x_master_send (parse_data.reg_num,parse_data.burst_array,parse_data.burst_size);
       } while (parse_data.current_loc != MINIDSP_PARSING_END);
}
```

Preliminary

```
/*
 *-------------------------------------------------------------------------
 * Function: minidsp_get_burst
 * Purpose:  construct burst array for miniDSP registers data.
 *-------------------------------------------------------------------------
 */

void minidsp_get_burst(const reg_value * program_ptr, int program_size, minidsp_parser_data * parse_data)
{
    int index = parse_data->current_loc;
    int burst_write_count = 0;

    /* check if first location is page register, and populate page addr */
    if (program_ptr[index].reg_off == 0){
        parse_data->page_num = program_ptr[index].reg_val;
        parse_data->reg_num = program_ptr[index].reg_off;
        parse_data->burst_array[burst_write_count++] = program_ptr[index].reg_val;
        index++;
        goto finish_out;
    }

    /* if it's not page register, store the reg_off and val into array */
        parse_data->reg_num = program_ptr[index].reg_off;
        parse_data->burst_array[burst_write_count++] = program_ptr[index].reg_val;
        index++;

    /* check if the reg addr is continue or not */
     for (; index < program_size; index++){
        if (program_ptr[index].reg_off != (program_ptr[index - 1].reg_off + 1)){
            break;
        }
        else{
            parse_data->burst_array[burst_write_count++] = program_ptr[index].reg_val;
        }
     }

finish_out:

        parse_data->burst_size = burst_write_count;
        if (index == program_size){
            /* parsing completed */
            parse_data->current_loc = MINIDSP_PARSING_END;
        }
        else{
            parse_data->current_loc = index;
        }
}
```

Preliminary

THIS PROGRAM IS PROVIDED "AS IS". TI MAKES NO WARRANTIES OR REPRESENTATIONS, EITHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, LACK OF VIRUSES, ACCURACY OR COMPLETENESS OF RESPONSES, RESULTS AND LACK OF NEGLIGENCE. TI DISCLAIMS ANY WARRANTY OF TITLE, QUIET ENJOYMENT, QUIET POSSESSION, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO THE PROGRAM OR YOUR USE OF THE PROGRAM.

IN NO EVENT SHALL TI BE LIABLE FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES, HOWEVER CAUSED, ON ANY THEORY OF LIABILITY AND WHETHER OR NOT TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, ARISING IN ANY WAY OUT OF THIS AGREEMENT, THE PROGRAM, OR YOUR USE OF THE PROGRAM. EXCLUDED DAMAGES INCLUDE, BUT ARE NOT LIMITED TO, COST OF REMOVAL OR REINSTALLATION, COMPUTER TIME, LABOR COSTS, LOSS OF GOODWILL, LOSS OF PROFITS, LOSS OF SAVINGS, OR LOSS OF USE OR INTERRUPTION OF BUSINESS. IN NO EVENT WILL TI'S AGGREGATE LIABILITY UNDER THIS AGREEMENT OR ARISING OUT OF YOUR USE OF THE PROGRAM EXCEED FIVE HUNDRED DOLLARS (U.S.$500).

Unless otherwise stated, the Program written and copyrighted by Texas Instruments is distributed as "freeware". You may, only under TI's copyright in the Program, use and modify the Program without any charge or restriction. You may distribute to third parties, provided that you transfer a copy of this license to the third party and the third party agrees to these terms by its first use of the Program. You must reproduce the copyright notice and any other legend of ownership on each copy or partial copy, of the Program.

You acknowledge and agree that the Program contains copyrighted material, trade secrets and other TI proprietary information and is protected by copyright laws, international copyright treaties, and trade secret laws, as well as other intellectual property laws. To protect TI's rights in the Program, you agree not to decompile, reverse engineer, disassemble or otherwise translate any object code versions of the Program to a human-readable form. You agree that in no event will you alter, remove or destroy any copyright notice included in the Program. TI reserves all rights not specifically granted under this license. Except as specifically provided herein, nothing in this agreement shall be construed as conferring by implication, estoppel, or otherwise, upon you, any license or other right under any TI patents, copyrights or trade secrets.

You may not use the Program in non-TI devices.

Preliminary