Assemble Language Guide

bd3201(DRE)的程序可以是机器码（如bd3201的datasheet里所述），也可以是如下的汇编语言。

1. 综述

bd3201的assembler不区分大小写，忽略空白行，多个tab或空格将会被当作是一个空格。每个指令占一行。

2. 地址格式

地址可以是十六进制或十进制的格式。如果用十六进制表示，需要在地址前加前缀“$”符或“0x”，或者加后缀 “H”。
例如，以下几条指令指向同一地址：

```
RZP $4          K=0.5       ；读地址4，乘以0.5
RZP 004H        K=0.5       ；读地址4，乘以0.5
RZP 4           K=0.5       ；读地址4，乘以0.5
RZP 0x04        K=0.5       ；读地址4，乘以0.5
```

3. 系数格式

系数可以是多种不同的数字格式，包括二进制，八进制，十六进制以及十进制。用二进制表示时，加前缀“%”或后缀“B”。八进制后缀为“Q”。十六进制加前缀“$”或“0x”，或加后缀 “H”。十进制不加符号，直接用数字表示。十进制数可以有小数点，也可以有正（+）负（−）号。bd3201的所有MAC指令系数存储在固定位置，符号-数值用S.7格式表示。其中S代表符号位，7代表小数部分的数值。当用二进制，八进制或十六进制表示时，数值格式为右对齐，整数和小数部分不分开，未定义的前几位补0。当用十进制表示时，数值格式不变，也可以加小数点与S.7格式数值一致。举例说明，下面所有表示方式都代表S.7格式的数值-0.34375：

```
Binary(二进制): %10101100          Hexadecimal（十六进制）: $AC
                10101100B                                   0xAC
                                                            ACH
Octal（八进制）: 254Q              Decimal（十进制）: -0.34375
                                                    -44
                                                    172
```

将十进制数转换为适用于编译器的十六进制，八进制或二进制数的一个简单方法是，移动二进制点至最右端使其恰为整数。例如，将数值0.1796875转为适合编译器的十六进制，八进制或二进制数：
乘以2^7 (因此可以变S.7为S7)：
$0.1796875 \times 2^7 = 0.1796875 \times 128 = 23 = 17H = 27Q = 10111B$
对于数值-0.1796875:
$-0.1796875 \times 2^7 = -0.1796875 \times 128 = -23 = 97H = 227Q = 10010111B$
*注意数值取8位。

4. LFO 设置指令格式

bd3201有四种低频振荡器(LFOs)是必须要在使用之前进行设置的。这些设置指令可以出现在汇编文件的任何地方。设置这些参数（振幅，频率，波形）需要LFO设置指令，格式如下：

LFOn WAVESHAPE AMP=a FREQ=f [XFAD=x] [; COMMENTS]
说明: 方括号 以内为可选。

通过设置参数n来选择LOFs 0-3。波形可以为“SIN”正弦，“TRI”三角波，或者“SAW”锯齿波。振幅系数AMP是一个15位无符号数（范围0-32767 或 $0-$7FFF），同时采样率为±AMP/8。频率系数FREQ是一个13位无符号数（范围0-8191 或 $0-$1FFF）。XFAD选择用于pitch转换算法的crossfade形状，此参数设置仅对锯齿波有效，有效的crossfade设置为：1, 1/2, 1/8, 1/16。如果要计算振荡器产生的频率，公式如下：
f(Sinusoids) = (F/M) Fs/(2)
这里
F = 13位频率系数FREQ
Fs = 取样率
M = 262143 (0x3FFFF, 内部最大值18位)
取样率为48kHz，则 f(Sinusoids) = 0.029142 F
f(Sinusoids) = (F/M) Fs/(2)

f(Triangles) = (F*Fs*C)/(4*M*SIN)
这里
F = 13位频率系数FREQ
Fs = 取样率
C = 4194304 (0x400000, LFO内部使用的常量)
M = 262143 (0x3FFFF, 内部最大值18位)
SIN = 8388607 (0x7FFFFF, 最大位置24位)
取样率为48kHz，则 f(Triangles) = 0.022888 F

F(Sawtooths) = 2 F(Triangles)
取样率为48kHz，则 f(Triangles) = 0. 045777 F
F(Sawtooths) = 2 F(Triangles)
For a sample rate of 48kHz, f(Triangles) = 0. 045777 F

举例说明：
LFO0 SIN AMP=0x3FFF FREQ=0x200；14.9Hz, 1/2 振幅正弦
LFO1 TRI AMP=10000 FREQ=3000；68.7Hz 三角波
LFO2 SAW AMP=$400 FREQ=512 XFAD=1/8；11.7Hz, 1/32 振幅锯齿波

5. Chorus 指令格式

Chorus 指令格式如下：
[CHRn] OPCODE ADDRESS [CHORUS CONTROLS] [; COMMENTS]
说明：方括号内为可选。

CHRn选择了chorus指令，n表示相关的LFO驱动指令。ADDRESS表明I/O（Address 0是左声道，Address 1是右声道），相关的数据RAM地址，或内存标记。CHORUS CONTROLS 是一个激活这些功能的无序参数列表。
OPCODE 控制了MAC的基本功能，它定义了读或写操作，选择哪个寄存器（空，Acc, B, 或 C）被累加到乘法器输出上。寄存器B能存储当前指令的被乘数，而寄存器C和Acc一样也能存储同样的值。

6. MEM 语句

编译器用MEM语句来声明地址标记并分配数据RAM的使用。地址标记可以放在程序的任何位置，只要在使用之前声明即可。MEM语句的格式如下：

MEM NAME SIZE [; COMMENTS]

这里NAME是一个字符串，必须以字母开头，不允许有空格。SIZE是内存区的大小，以十进制或十六进制来表示。用十六进制表示时，须加前缀“0x”或后缀“H”。内存大小也可以被定义成毫秒（假设取样率44.1kHz），须在数字后面加“ms”。

例如：

```
MEM LPF        1          ；定义一个延时单元，长度为1个样点
MEM delay      100        ；定义一个100个样点长度的延时
MEM FIR        0x20       ；定义一个32个样点长度的延时
MEM predelay 50.0ms   ；定义一个50.0ms的延时
```

编译器将在Address 0分配起始数据RAM并工作至高地址位。

说明 由于定义声明要占据内存偏移计数器，所以实际的内存位置要比分配的地址大1。还要注意由于内存偏移计数器是向下计数的，向低位地址写入的数据要有延时地从高位地址中读取。

例如:

```
WZP 0H  K=0.5 ；写入将被延时的数据
RZP  4H  K=0.5 ；读延时了4个样点的数据
```

一旦你已经定义了内存单元，你可以有很多种方法访问它们。最简单的就是用指向该延时行的名字。后缀单引号(')指向行尾，后缀双引号(")指向行中。

例如，以下是延时100个样点的程序代码：

```
MEM       delay     100          ；定义了100个样点的延时
RZP        INL      127          ；得到左声道输入值传给累加器
WAP       delay     0            ；向延时行头写入累加器
RZP       delay'    127          ；从延时行尾读出数据
WAP       OUTL   0               ；将累加器的数据写入左声道输出
```

你可以用（+）或（-）符号来定义延时行的偏移量。

例如，访问一个FIR滤波器的第512个头：

```
MEM       FIR        1023       ;使用数据RAM的1023个位置
RZP        INR        127       ;得到右声道输入给累加器
WAP       FIR        0          ;向延时行头写入累加器
RZP       FIR+512    127        ;读滤波器的第512个头
WAP      OUTR        0          ;将累加器的数据写入右声道输出
```

在以调指令里可以既有后缀也可以有偏移量。

7. ABS 语句 Statement

编译器使用ABS语句来声明绝对地址标记。ABS语句的格式如下：

ABS LABEL [ADDRESS] [; COMMENTS]

这里LABLE是一个字符串，必须以字母开头，不允许有空格。ADDRESS是绝对地址。地址必须以十进制或十六进制整数来定义。十六进制表示时要加前缀“0X”或后缀“H”。例如：

```
ABS param 0x0081 ；使用内存地址作为参数
```

内存地址偏移量以及129以上的地址不用ABS语句描述。只有130以下的地址可以用ABS访问。
例如
ABS hparams 0x0082；这将导致出错
ABS params 0x0081 ；正确
RZP params+3 0.5 ；这将导致出错
RZP params 0.5 ；正确

8. EQU 语句

编译器用EQU语句来定义系数常量。EQU语句格式如下：
NAME EQU VALUE [; COMMENTS]
这里NAME是一个字符串，必须以字母开头，不允许有空格。VALUE是表示NAME的一个数字，必须符合系数格式。
例如:
gain EQU 0.5 ；定义增益系数-6dB
RZP INL gain ；得到累加器中的左声道输入，增益下降
WAP OUTL 0.0；将累计器写至左声道输出
注意EQU语句是非递归的。你不能使一个名字等于一个数值，也不能使一个名字等于另一个名字。

9. 读指令

对读指令操作，Product = K*[ADDRESS]，因此寄存器B可以通过地址或标记来存储左声道输入，右声道输入，数据RAM值。
RZP Read, Acc = Zero + Product
RAP Read, Acc = Acc + Product
RBP Read, Acc = B Register + Product
RCP Read, Acc = C Register + Product
RZPB Read, Acc = Zero + Product；加载B寄存器
RAPB Read, Acc = Acc + Product；加载B寄存器
RBPB Read, Acc = B Register + Product；加载B寄存器
RCPB Read, Acc = C Register + Product；加载B寄存器
RZPC Read, Acc = Zero + Product；加载C寄存器
RAPC Read, Acc = Acc + Product；加载C寄存器
RBPC Read, Acc = B Register + Product；加载C寄存器
RCPC Read, Acc = C Register + Product；加载C寄存器
RZPBC Read, Acc = Zero + Product；加载B和C寄存器
RAPBC Read, Acc = Acc + Product；加载B和C寄存器
RBPBC Read, Acc = B Register + Product；加载B和C寄存器
RCPBC Read, Acc = C Register + Product；加载B和C寄存器

10. 写指令

对写指令操作，Product = k*Acc ，因此寄存器可以从上一个tick中存储Acc的值。写入内存存储了前一个指令的结果，写向输出端传输了当前指令的结果。
WZP Write, Acc = Zero + Product
WAP Write, Acc = Acc + Product
WBP Write, Acc = B Register + Product
WCP Write, Acc = C Register + Product

WZPB Write, Acc = Zero + Product；加载B寄存器
WAPB Write, Acc = Acc + Product；加载B寄存器
WBPB Write, Acc = B Register + Product；加载B寄存器
WCPB Write, Acc = C Register + Product；加载B寄存器
WZPC Write, Acc = Zero + Product；加载C寄存器
WAPC Write, Acc = Acc + Product；加载C寄存器
WBPC Write, Acc = B Register + Product；加载C寄存器
WCPC Write, Acc = C Register + Product；加载C寄存器
WZPBC Write, Acc = Zero + Product；加载B和C寄存器
WAPBC Write, Acc = Acc + Product；加载B和C寄存器
WBPBC Write, Acc = B Register + Product；加载B和C寄存器
WCPBC Write, Acc = C Register + Product；加载B和C寄存器
例如:

```
RZP     delay”       K=.500；读（中间延时）/2入Acc
WAP   delay+125 192；写入第125个延时点，Acc = -0.5*Acc
RAPB  delay'        K=128；  Acc = -1*(延时尾部)+Acc, B = 延时尾部
WBPC delay’-1          ；写入倒数第二个延时点，  Acc & C = B + 0*Acc
RCP     ADCL      K=0x7F；读~1*(左声道输入)到Acc和C
WCPB OUTR       K=-0x40；B=Acc，写C-0.5*Acc入有声道输出和Acc
```

11. Chorus 控制指令

chorus语句定义了chorus命令指令。他们重新定义了系数字的代替含义，以及chorus指令中缺少的系数字。为了在两个整数取样率之间插入一个小数部分，两个相邻的指令用来生成chorus。
Chorus控制指令由以下参数组成：
The Chorus Controls consist of the following parameters:
[+SIN|-SIN|+COS|-COS] [COMPK] [COMPA] [MASKA] [LATCH] [COMPS]
+SIN 使用所选LFO的正弦正值输出
-SIN 使用所选LFO的正弦负值输出
+COS 使用所选LFO的余弦正值输出
-COS 使用所选LFO的余弦负值输出
每一个LFO有4个点：正负正弦和正负余弦输出。上面的参数说明了使用哪一个点。如果没有定义，则默认值为+SIN，符号是必需的，不写符号只一个“SIN”会报错不能通过编译。

COMPK　　chorus插补系数的1’s补码。
LFO的输出分为两部分：偏移量地址和插补系数。偏移量地址被发送至数据RAM地址发生器，且用户不能访问。插补系数被用于指定地址的值，系数的1’s补码用于地址+1的值。COMPK参数体现了系数的1’s补码。

COMPA　chorus插补地址的1’s补码。
这个参数体现了chorus插补地址的1’s补码，通过改变相位可以180°改变chorus的波形。定义COMPA为“+ SIN”就等于定于为“- SIN”。实际上在COMPA里“- SIN”和“- COS”也是相同的定义。这个语句用于pitch变换以增加声音的透明度。适当使用的LFO输出语句将回导致这个命令多余。

MASKA　　屏蔽chorus发生器偏移地址，选择crossfade 系数。

当进行pitch转换所需要的两个插补数完成之后，他们必须从内存中被读回，并计算crossfade系数总和。在chorus命令中，地址发生器产生绝对地址时要将chorus偏移地址加上，这会产生MASKA参数的覆盖，因此偏移量和从指令中特定读出的地址要被屏蔽。这个参数还定义了LFO设置指令所用的crossfade系数。

LATCH　　锁住所选LFO的数据。
当一对chorus命令实现后，LFO数据必须被锁住，否则，LFO的值在命令执行的时候就会改变，会出现不可预料的错误结果。这个指令出现在第一个chorus命令后第二个chorus前。

COMPS　　chorus 偏移地址符号位的1's补码。
这个参数体现了chorus 偏移地址符号位的1's补码。这是前一个IC版本修订的剩余物，用在这里表示信息目的。
例如:
CHR0 RZP mem -COS COMPK LATCH ； 读mem小数部分的-COS chorus
CHR0 RAP mem+1 -COS ; 将第2个地址计入总和，包括小数部分
CHR1 RZP stor1 COMPK MASKA LATCH ; 读stor 1的crossfade的小数部分
CHR1 RAP stor2 MASKA ; 将stor 2 的crossfade 小数部分计入总和

12. 保留内存标记

为了方便实用，一些内存标记已经被编译器预先定义，访问它们与访问其他内存单元一样，它们有：
INL ---------- Left Input (read only)左声道输入（只读）
ADCL ---------- Left Input (read only) 左声道输入（只读）
INR ---------- Right Input (read only) 右声道输入（只读）
ADCR ---------- Right Input (read only) 右声道输入（只读）
OUTL ---------- Left Output (write only) 左声道输出（只写）
OUTR ---------- Right Output (write only) 右道输出（只写）
例如:

```
RZP     INL       0.5     ; Read Left Input读左声道输入
RZP     ADCL      0.5     ; Read Left Input读左声道输入
RZP     0         0.5     ; Read Left Input读左声道输入
```

13. 注释

注释语句前要加“；”，编译器将忽略分号后的整行所有字符。
例如：

```
; ::::::::This is a valid comment::::::::
RZP mem $5 ;This is also a valid comment; all text here is ignored.
```

bd3201 Assemble Language Guide（中文版）

www.bdnc.com

香港公司

比特联创(香港)有限公司
香港沙田科技大道西 6 号集成电路开发中心 512-513 室

电话: 852-28542731/ 23916797
传真: 852-23916796
电子邮件: general@bdnc.com

北京公司

比特联创电子(北京)有限公司
北京市海淀区三里河路 21 号甘家口大厦写字楼南座 1513 室，邮编：100037

电话: 86-10-88392985/88392986
传真: 86-10-88392980
电子邮件: bdncele@public.bta.net.cn