MCU 与 SHARC 的 VisualAudio 控制协议 v0.6

目录

注：在 v0.4 基础上添加“ID=8：控制 32*32 混音器”，相应的代码如下：

```
void SetMixer(u32 *pCmd)// id=8
{
    int outch, i, inamp[20], amp;
    outch = pCmd[0];// 0:31-->out1:out32
    for(i=0; i<8; i++)//总共有 8*4=32 个控制字节
    {
        amp = pCmd[i+1];
        inamp[i*4] = amp>>24;
        amp>>=8;
        inamp[i*4+1] = amp>>24;
        amp>>=8;
        inamp[i*4+2] = amp>>24;
        amp>>=8;
        inamp[i*4+3] = amp>>24;
    }

    for(i=0; i<32; i++)//有 32 个输入
        Layout1_MixerNxM32x32amps[outch + 32*i]      = inamp[i]/100.0; //有 32 个输出
}
```

MCU 发给 DSP 协议格式：（每个单元由一个字（4 个字节）组成），注意：发送时低字节先发

ID=0：控制所有模块的状态

帧头	ID/长度	首地址	数据(int)	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	模块地址	模块状态： 0:无效;1:有效; 2:静音 3:旁路 整型	长度+首地址+数据长度+具体数据之和

例如：要改变 ScalerCH1_1 模块的状态(_Layout1_ScalerCH1_1 = 0x000C056A)为静音，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头 发送按照 EF BE AD DE 低字节先发送
0x0004 0000 //长度=4; ID=0
0x000C056A //模块地址
0x000000002 // 2:静音
0xxxxxxxxx //CRC 校验=0x00040000+ 0x000C056A +0x2

ID=1：控制噪声门

帧头	ID/长度	首地址	threshold	attackTime	decayTime	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	噪声门模块地址;	-120---0 dB 数据类型： 浮点	20---1000ms 浮点	20---1000ms 浮点	ID/长度+首地址+数据长度+具体数据之和

例如：要设置 NoiseGate1 模块 (_Layout1_NoiseGate1=0x000C0500)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x01060000 //长度=6; ID=1
0x000C0500 // 模块地址
0xC1200000 // threshold = -10; 因为 threshold 的类型是浮点，发给 DSP 的也要是 32 位表示的浮点数
0x 41A00000 // attackTime = 20.0
0x 41A00000 // decayTime = 20.0
0xxxxxxxxx //CRC 校验=0x01040000+0x000C0500 +0xC1200000+0x 41A00000+0x 41A00000

DSP 返回 MCU 协议格式：

帧头	长度<<16	NoiseGateOn	CRC
0xDEADBEEF	0x00030004	0：噪声门关闭 1：噪声门开启	ID/长度+处理结果之和

例如： 0xDEAD BEEF // 帧头
0x00030004 //长度=3， 固定值 4
0x00000001 // NoiseGateOn = 1
0xxxxxxxxx //CRC 校验同上

ID=2：控制移频

帧头	ID/长度	移频量	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	1---10Hz 整型	ID/长度+首地址+数据长度+具体数据之和

例如：要设置移频量是 1Hz，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送

0x02030000 //长度=3; ID=2

0x00000001 // 移频量

0xxxxxxxxx //CRC 校验同上

ID=3：控制高低通滤波器

帧头	ID/长度	首地址	HLPF &Type& Slope	freq	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	高低通滤波器模块地址;	HLPF: 1---2 Type:0—2; Slope:0---6 整型	10---20k Hz 浮点	ID/长度+首地址+数据长度+具体数据之和

注：HLPF: 1= HPF; 2= LPF；位于低 8 位

Type: 0: Bessel; 1:Butterworth; 2:Link-Riley 位于高 8 位

Slope: 0：-6dB 1：-12dB 2：-18dB 3：-24dB 4：-36dB 5：-48dB 位于高 16 位

例如：要设置 HighPassCH1 模块（_Layout1_HighPassCH1=0x000C052F)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头

0x03050000 //长度=5; ID=3

0x000C0500 // 模块地址

0x00030201 // HLPF =1:HPF; Type =2: Link-Riley; Slope=3：-24dB

0x 447A0000 // freq=1000.0

0xxxxxxxxx //CRC 校验同上

ID=4：控制 PEQ

帧头	ID/长度	首地址	Ch&Type& Band	freq	gain	Q	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len= 除帧头外，所有字的个数	PEQ 模块地址	**Ch :0---31** **Type:3---5** **Band =0---7** 整型	10---20000Hz 浮点	-20---20dB 浮点	0.5---20 浮点	ID/长度+首地址+数据长度+具体数据之和

注：Ch=0---31: 0~15: 输入通道 1~16; 16~31: 输出通道 1~16， 位于低 8 位

Type=3---5: 3=PEQ; 4= lowShelf; 5= highShelf， 位于高 8 位

Band =0---30，共 31 段，（注：前面只有 0~14: 共 15 段)， 位于高 16 位

Gain = 0 时，表示此段 PEQ Bypass

例如：要设置 PEQ5BandCH1 (_Layout1_PEQ5BandCH1=0x000C0538)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头
0x04070000 //长度=7; ID=4
0x000C0538 // 模块地址
0x00010302 // Ch=2; Type=3= PEQ; Band =1;
0x 447A0000 // freq=1000.0
0x 00 // gain = 0.0
0x 3F800000 // Q = 1.0
0xxxxxxxxx //CRC 校验同上

ID=5：控制压限

帧头	ID/长度	首地址	Threshold	gain	Knee Depth	ratio	Attack Time	Decay Time	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len= 除帧头外，所有字的个数	压限模块地址;	-120---0 dB 浮点	0---100 浮点	0.1---60 浮点	1---100 浮点	20---1000ms 浮点	20---1000ms 浮点	ID/长度+首地址+数据长度+具体数据之和

例如：要设置 Limiter1 模块 (_Layout1_AGCLimiterCore1=0x000C0554)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x05090000 //长度=9; ID=5
0x000C0554 // 模块地址
0xC1200000 // threshold = -10;
0x00000000 // gain = 0;
0x3F800000 // kneedepth = 1;
0x3F800000 // ratio= 1;
0x 41A00000 // attackTime = 20.0
0x 41A00000 // decayTime = 20.0
0xxxxxxxxx //CRC 校验同上

ID=6：控制 scale 电平大小

帧头	ID/长度	首地址	gain	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	电平模块地址;	-100~0 浮点	ID/长度+首地址+数据长度+具体数据之和

例如：要设置 ScalerCH1_1 模块 (_Layout1_ScalerCH1_1= 0x000C056A)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x06040000 //长度=4; ID=6
0x000C056A // 模块地址
0x 3F800000 // gain =1
0xxxxxxxxx //CRC 校验同上

ID=7：控制电平表

帧头	ID/长度	首地址	attackTime	decayTime	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	电平表模块的地址;	3---1000ms 浮点	3---1000ms 浮点	ID/长度+首地址+数据长度+具体数据之和

例如：要设置 ScalerCH1_1 模块（_Layout1_CH1meter=0x000C05C2)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送

0x07050000 //长度=5; ID=7

0x000C05C2 // 模块地址

0x 41A00000 // attackTime = 20.0

0x 41A00000 // decayTime = 20.0

0xxxxxxxxx //CRC 校验同上

DSP 返回 MCU 协议格式：

帧头	长度<<16	Cur	Pk	CRC
0xDEADBEEF	0x00040004	浮点 单位：dB	浮点 单位：dB	ID/长度+处理结果之和

例如： 0xDEAD BEEF // 帧头

0x00040004 //长度=4， 固定值 4

0x3F800000 // Cur = 1.0

0x3F800000 // Pk = 1.0

0xxxxxxxxx //CRC 校验同上

ID=8：控制 18*16 混音器

帧头	ID/长度	outch	Value1	Value2	Value3	Value4	Value5	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len= 除帧头外，所有字的个数	Ch1---ch16: 0---15 整型	整型	整型	整型	整型	整型	ID/长度+首地址+数据长度+具体数据之和

注：Value1 = v1<<24 | v2<<16 | v3<<8 | v4;

Value2 = v5<<24 | v6<<16 | v7<<8 | v8;

Value3 = v9<<24 | v10<<16 | v11<<8 | v12;

Value4 = v13<<24 | v14<<16 | v15<<8 | v16;

Value5 = v17<<24 | v18<<16;

某个输出通道 Outch 的电平 = v1*in1 + v2*in2 + … + v17*in17 + v18*in18

v1---v18 分别表示 18 个输入通道的电平值

发送时 v*100，比如界面上混音值是 0.05，则发送值是 5

则混音的最大值只能是 2.55（乘以 100 后，就是 255），

注：此协议只支持 18*16 的 Mixer 模块，如要控制输入输出数不一样的 Mixer，需要修改协议

例如：要设置混音器，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x08080000 //长度=8; ID=8
0x 0 // outch = 0
0x010001000 // v1=1; v2=0; v3=1; v4=0
0x 010001000 // v5=1; v6=0; v7=1; v8=0
0x 010001000 // v9=1; v10=0; v11=1; v12=0
0x 010001000 // v13=1; v14=0; v15=1; v16=0
0x 010000000 // v17=1; v18=0;
0xxxxxxxxx //CRC 校验同上

ID=8：控制 32*32 混音器

帧头	ID/长度	outch	Value 1	Value 2	Value 3	Value 4	Value 5	Value 6	Value 7	Value 8	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len= 除帧头外，所有字的个数	Ch1---ch32: 0---31 整型	整型	整型	整型	整型	整型	整型	整型	整型	ID/长度 + 首地址 + 数据长度 + 具体数据之和

注：Value1 = v1<<24 | v2<<16 | v3<<8 | v4;
Value2 = v5<<24 | v6<<16 | v7<<8 | v8;
Value3 = v9<<24 | v10<<16 | v11<<8 | v12;
Value4 = v13<<24 | v14<<16 | v15<<8 | v16;
Value5 = v17<<24 | v18<<16 | v19<<8 | v20;
Value6 = v21<<24 | v22<<16 | v23<<8 | v24;
Value7 = v25<<24 | v26<<16 | v27<<8 | v28;
Value8 = v29<<24 | v30<<16 | v31<<8 | v32;
某个输出通道 Outch 的电平 = v1*in1 + v2*in2 + … + v31*in31 + v32*in32
v1---v32 分别表示 32 个输入通道的电平值
发送时 v*100，比如界面上混音值是 0.05，则发送值是 5
则混音的最大值只能是 2.55（乘以 100 后，就是 255），
注：此协议只支持 32*32 的 Mixer 模块，如要控制输入输出数不一样的 Mixer，需要修改协议

例如：要设置混音器，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x08080000 //长度=8; ID=8
0x 0 // outch = 0
0x010001000 // v1=1; v2=0; v3=1; v4=0
0x 010001000 // v5=1; v6=0; v7=1; v8=0
0x 010001000 // v9=1; v10=0; v11=1; v12=0
0x 010001000 // v13=1; v14=0; v15=1; v16=0
0x 010001000 // v17=1; v18=0; v19=1; v20=0
0x 010001000 // v21=1; v22=0; v23=1; v24=0
0x 010001000 // v25=1; v26=0; v27=1; v28=0
0x 010001000 // v29=1; v30=0; v31=1; v32=0
0xxxxxxxxx //CRC 校验同上

ID=9：控制延时

帧头	ID/长度	首地址	delay	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	延时模块地址;	1.34—maxDelay ms 浮点	ID/长度+首地址+数据长度+具体数据之和

例如：要设置 DelayOffChipCH1 模块（_Layout1_DelayOffChipCH1=0x000C05B2），需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x09040000 //长度=4; ID=9
0x000C05B2 // 模块地址
0x41A00000 // delay = 20ms
0xxxxxxxxx //CRC 校验同上

ID=10：控制正弦波发生器频率

帧头	ID/长度	首地址	频率	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	正弦模块地址;	0---24k 浮点	ID/长度+首地址+数据长度+具体数据之和

例如：要设置 TestGenSine 模块（_Layout1_TestGenSine=0x000B4208）频率为 1000Hz，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x0a040000 //长度=4; ID=a
0x000B4208 // 模块地址
0x447A0000 // 1000Hz
0xxxxxxxxx //CRC 校验同上

ID=11：往任意地址里面写任意值

帧头	ID/长度	首地址	具体数据	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	音频模块的参数地址;	具体写入什么数据	ID/长度+首地址+数据长度+具体数据之和

例 1：要把 level 模块的 amps (_Layout1_levelamps=0x000B88DC)设成 1.0，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送

0x0B040000 //长度=4; ID=11

0x000B88DC // amps 参数地址

0x3F800000 // 1.0；因为 amp 的类型是浮点，所以发给 DSP 的也要是 32 位表示的浮点数

0x 3F8F88DC //CRC 校验=0x00040000+0x000B88DC +0x3F800000

ID=12：从任意地址里面读任意值

帧头	ID/长度	地址	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	模块参数的地址	长度+首地址+数据长度+具体数据之和

例如：要读取 level 模块的当前值(_Layout1_levelamps=0x000B88DC)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头

0x0C03 0000 //长度=3; ID=12

0x000B88DC //参数地址

0xxxxxxxxxx //CRC 校验

DSP 返回 MCU 协议格式：

帧头	长度<<16	返回数据	CRC
0xDEADBEEF	0x00030004	浮点	ID/长度+处理结果之和

例如： 0xDEAD BEEF // 帧头

0x00030004 //长度=3， 固定值 4

0x3F800000 //返回数据= 1.0

0xxxxxxxxx //CRC 校验同上

ID=13：批量读取电平表，同时也可以控制所有的电平表

帧头	ID/长度	attackTime	decayTime	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	3---1000ms 浮点	3---1000ms 浮点	ID/长度+首地址+数据长度+具体数据之和

例如：要设置 ScalerCH1_1 模块（_Layout1_CH1meter=0x000C05C2)，需要发送给 DSP 的指令如下：

0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送

0x0D040000 //长度=4; ID=13

0x 41A00000 // attackTime = 20.0

0x 41A00000 // decayTime = 20.0

0xxxxxxxxx //CRC 校验同上

DSP 电平值批量返回如下：

帧头	长度<<16	D1	D2	D3	D3	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
0xDEADBEEF	0x00220004																

D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28	D29	D30	D31	D32	CRC
																ID/长度+首地址+数据长度+具体数据之和

每通道用 16 位数据表示：

CH1meter1：D1>>16 （用 D1 高 16 位表示）

CH2meter1：D1&0xFFFF （用 D1 低 16 位表示）

……

CH31meter1：D16>>16 （用 D16 高 16 位表示）

CH32meter1：D16&0xFFFF （用 D16 低 16 位表示）

CH1meter2：D17>>16 （用 D17 高 16 位表示）

CH2meter2：D17&0xFFFF （用 D17 低 16 位表示）

……

CH31meter2：D32>>16 （用 D32 高 16 位表示）

CH32meter2：D32&0xFFFF （用 D32 低 16 位表示）

获取到数据后，全部利用以下公式计算 meter 最终的增益：

比如 CH1meter1：

CH1meter1 = CH1meter1 / 10.0 – 80; // meter 范围： -80 到+24dB

ID=14：批量读取 scale 电平大小，同时也可以控制所有的 scale

帧头	ID/长度	gain	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	-100~0 浮点	ID/长度+首地址+数据长度+具体数据之和

注意：如果 gain=0xffffffff 则表示只是读取 scale 大小，不控制 scale，不改变原 scale 任何参数

例如：要设置 ScalerCH1_1 模块（_Layout1_ScalerCH1_1= 0x000C056A)，需要发送给 DSP 的指令如下：
0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x0E030000 //长度=3; ID=14
0x 3F800000 // gain =1
0xxxxxxxxx //CRC 校验同上

DSP scale 电平大小批量返回如下：

帧头	长度<<16	D1	D2	D3	D3	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
0xDEADBEEF	0x00220004																

D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28	D29	D30	D31	D32	CRC
																ID/长度+首地址+数据长度+具体数据之和

每通道用 16 位数据表示：
ScalerCH1：D1>>16 （D1 高 16 位）
ScalerCH2：D1&0xFFFF （D1 低 16 位）
……
ScalerCH31：D16>>16 （D16 高 16 位）
ScalerCH32：D16&0xFFFF （D16 低 16 位）
ScalerMixerPosrCH1：D17>>16 （D17 高 16 位）
ScalerMixerPosrCH2：D17&0xFFFF （D17 低 16 位）
……
ScalerMixerPosrCH31：D32>>16 （D32 高 16 位）
ScalerMixerPosrCH32：D32&0xFFFF （D32 低 16 位）

获取到数据后，全部利用以下公式计算 Scaler 最终的增益：
比如 ScalerCH1：
ScalerCH1= ScalerCH1/ 10.0 – 100; // meter 范围： -100 到+12dB

ID=15：批量读取噪声门状态，同时也可以控制所有的噪声门

帧头	ID/长度	threshold	attackTime	decayTime	CRC
0xDEADBEEF	ID<<24 | Len<<16 Len=除帧头外，所有字的个数	-120---0 dB 数据类型：浮点	20---1000ms 浮点	20---1000ms 浮点	ID/长度+首地址+数据长度+具体数据之和

注意：如果 threshold =0xffffffff 则表示只是读取噪声门状态，不控制噪声门，不改变原噪声门的任何参数

例如：要设置 NoiseGate1 模块（_Layout1_NoiseGate1=0x000C0500)，需要发送给 DSP 的指令如下：
0xDEAD BEEF // 帧头，发送按照 EF BE AD DE 低字节先发送
0x0F050000 //长度=5; ID=15
0xC1200000 // threshold = -10; 因为 threshold 的类型是浮点，发给 DSP 的也要是 32 位表示的浮点数
0x 41A00000 // attackTime = 20.0
0x 41A00000 // decayTime = 20.0
0xxxxxxxxx //CRC 校验=0x01040000+0x000C0500 +0xC1200000+0x 41A00000+0x 41A00000
DSP 噪声门状态批量返回如下：

帧头	长度<<16	DATA	CRC
0xDEADBEEF	0x00030004		ID/长度+首地址+数据长度+具体数据之和

每通道用 1 位数据表示：
NoiseGate1： DATA >>31 （DATA 最高位）
……
NoiseGate32：DATA &0x1 （DATA 最低位）

普通 ID 数据返回格式：

DSP 返回 MCU 协议格式：

帧头	长度<<16	DSP 处理结果	CRC
0xDEADBEEF	0x00030004	0---2	ID/长度+处理结果之和

例如： 0xDEAD BEEF // 帧头
0x00030004 //长度=3， 固定值 4
0x0 // DSP 处理结果 = 0：没错误
0xxxxxxxxx //CRC 校验同上

DSP 处理结果：
0： NO_ERROR
1： CRC_ERROR
2： NOT_HANDLED