VT03 Program Sound Generator Users Manual

The appendix of FMDemo.rar is 6502 Source Code，midi2vt.rar is the tool for midi transfer to VT03 asm form ，You can use Cakewalk to make the midi document。

Operation Introduction：

1、Use Cakewalk to make the midi file. Be care of that the maximum path is five for midi document. Separately name the correspondence are "Square1","Square2","Triangle","Noise","DPCM"。Specifically may refer to the appendix midi02.mid of midi2vt.rar。(Only can use Midi type2) The wants transformation to midi documents have 5 sounds axles, the transformation procedure will according to the high to low sequence corresponding to VT03's Square1 ,Square2 ,Triangle , Noise and DPCM, If this musical parts does not use then spatial sound axles, The transformation midi document was based on NTSC speed matching, if your system are PAL system then you have to adjust the music speed by manual.

2、Use midi2vt.exe to transfer the midi file, which you have been edited it by the Cakewalk. It can create bin file for immediately broadcast and it also create asm and h file for your edit。

3、About how to increase the asm file, please you refer 2MusData.asm of FMDemo.rar. In the meantime, this 3EffData.asm of FMDemo.rar includes fifty general timbres and you can use the joystick to select it and broadcast. Its code data may for the user reference use，Please you refer 5Music.asm to learn the driving program of broadcast，the relative introduction are as below：

==========

Procedure variable explanation

==========

>>> Music_Flag1 : The music broadcast procedure alternately symbolized1 with the user program. Mainly use in sound picture synchronization, when each music broadcast establishes zero.

>>> Music_Flag2 : The music broadcast procedure alternately symbolized2 with the user program. Be must insert the user program control.

>>> Music_Tone : Sound basis value，when entire music procedure initialization establishes zero.

>>> Music_MuteFlag : Static sound symbol，It was established by BIOS

bit 0 Sound effect static sound symbol
0 = Normal
1 = Static sound
bit 1-6 Null
bit 7 Music static sound symbol
0 = Normal
1 = Static sound

>>> System_Flag : System synthesis symbol

bit 0 Running status
0 = Normal
1 = Demo
bit 7 Television service pattern
0 = NTSC
1 = PAL

>>> Music_SpeedCount : The music broadcast speed decrease progressively the counter

>>> Music_SpeedIndex : Music speed index

>>> Music_SpeedValue : Music speed value, it was composed by 4 bytes, by broadcast procedure in turn index use.

>>> Music_SquareIndexTable : Pre-placed Square-wave data index table

---

>>> Music_TriangleIndexTable : Pre-placed Triangle data index table

---
>>> Music_NoiseDataTable : Pre-placed Noise data table

>>> Music_DPCMIndexTable : Pre-placed DPCM data index table

>>> Music_PlayBuffer ds 5*8 : Music broadcast data sheet

>>> Music_PlayControl1 : Music broadcast control register1
bit 0-6 Metre waiting counter
0 = 128
bit 7 the group of control data processing condition
0 = Disable
1 = Enable

>>> Music_PlayControl2 : Music broadcast control register2
bit 0-3 Current volume establishment
bit 4-5 Music cycle counter

>>> Music_PlayAddress : Current music control data address

>>> Music_TimbreAddress : Current timbre data address

>>> **Music_ToneBuffer ds 8*10 : Timbre output control**

>>> Music_ToneControl1 : Timbre output control register1
bit 0-6 Sound track waiting counter
0 = 64
bit 5 Timbre output status
0 = Normal
1 = Index
bit 7 This sound track processing status
0 = Disable
1 = Enable

>>> Music_ToneControl2 : Timbre output control register4
bit0-3 Circulation counter
bit4-7 Index output Mask value

>>> Music_ToneControl3 : Timbre output control register2
bit 0-5 Output gap value
bit 6,7 Index processing status
00 = Index output
01 = The Volume index adds the output
10 = The Note index adds the output
11 = The Volume and note index adds output

>>> Music_ToneControl4 : Timbre output control register3
bit 0-3 Index output/input counter
bit4-5 Index input Mask value

>>> Music_ToneAddress : Timbre control data address

>>> Music_ToneOutData0 : Output data0，Correspondence $40x0

>>> Music_ToneOutData1 : Output data1，Correspondence $40x1

>>> Music_ToneOutData2 : Output data2，Correspondence $40x2

>>> Music_ToneOutData3 : Output data3，Correspondence $40x3

==========

Procedure function explanation

==========

>>> MusicInitial : Music procedure initialization

Intake conditions： Music_AddrL,H Directional music data index table

Music data index table form：

adr Music index table address
adr Sound effect index table address
adr Square1 musical part sound effect index table address
adr Square2 musical part sound effect index table address
adr Triangle musical part sound effect index table address
adr Noise musical part sound effect index table address
adr DPCM musical part sound effect index table address

Music index table form：

adr The first music data address
adr The second music data address
... .
adr The Nth music data address

Music data form：

byt s1,s2,s3,s4 Music speed value
adr Square1 musical part music data
0 = Has not corresponded the data
Other normal
adr Square2 musical part music data
0 = Has not corresponded the data
Other normal
adr Triangle musical part music data
0 = Has not corresponded the data
Other normal
adr Noise musical part music data
0 = Has not corresponded the data
Other normal
adr DPCM musical part music data
0 = Has not corresponded the data
Other normal

Sound effect index table form：

adr The first sound effect data address
adr The second sound effect data address
... .
adr The Nth sound effect data address

Sound effect data form：

adr Square1musical part sound effect data
0 = Has not corresponded the data

Other normal
adr Square2 musical part sound effect data
0 = Has not corresponded the data
Other normal
adr Triangle musical part sound effect data
0 = Has not corresponded the data
Other normal
adr Noise musical part sound effect data
0 = Has not corresponded the data
Other normal

>>> MusicPlayEnable : Music broadcast

>>> MusicPlayDisable : Music static sound

>>> MusicEffectEnable : Sound effect broadcast

>>> MusicEffectDisable :Sound effect static sound

>>> MusicPlay :Music broadcast processing
Intake conditions:
REG A = Music serial number

>>> MusicEffect : Sound effect broadcast processing
Intake conditions:
REG A = Sound effect serial number

==========

Broadcast control data form：MPCC....Music broadcast Control Code

==========

>>> MPCC_Wait : Waits for the metre
Square/Triangle/Noise/DPCM musical part
db %0ccccccc
c = Waiting metre number(0-127)

>>> MPCC_Timbre : Establishment timbre
Square/Triangle musical part
db %1000iiii
i = Pre-placed timbre index
Noise/DPCM sound
Invalid

>>> MPCC_Volume : Establishment volume
Square/Noise musical part
db %1001vvvv
v = Volume
Triangle/DPCM musical part
Invalid

>>> MPCC_Inside : According to pre-placed data broadcast
Square/Triangle musical part
db %1010tttt , %cccccllll
t = Note
$0 1

$1 1#
$2 2
$3 2#
$4 3
$5 4
$6 4#
$7 5
$8 5#
$9 6
$A 6#
$B 7
$C spatial note
$D spatial note
$E spatial note
$F spatial note
l = treble
0 -...
1 -..
2 -.
3 =
4 +.
5 +..
6 null
7 null
c = Metre length
0-31
Noise & DPCM musical part
%1010iiii
i = Pre-placed timbre index(0-15)

>>> MPCC_Loop : Is assigning the address to the current address cycle broadcast
Square/Triangle/Noise/DPCM musical part
dw Address
c = Cycle-index
Address Assigns circulation address（May not the nesting）

>>> MPCC_ToneSet : Establishment sound basis value
Square/Triangle/Noise/DPCM musical part，Generally use in the Square musical part
db %1100tone
tone = New sound basis value

>>> MPCC_FlagSet :Establishment Music_Flag1 as data
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110000,data
When music broadcast, the Music_Flag is clean to zero.

>>> MPCC_FlagInc :Music_Flag1 adds one
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110001

>>> MPCC_FlagDec :Music_Flag1 adds one
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110010

>>> MPCC_FlagBne :If the Music_Flag1 is not equal to data then jumps the extension
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110011,data
dw Address
Address = Transfer address

>>> MPCC_FlagBeq :If the Music_Flag1 is equal to data then jumps the extension
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110100,data
dw Address
Address = Transfer address

>>> MPCC_FlagBneCall :If the Music_Flag1 is not equal to data then carries out exterior procedure
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110101
dw Address
Address = Exterior address on the program

>>> MPCC_FlagBeqCall : If the Music_Flag1 is equal to data then carries out exterior procedure
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110110
dw Address
Address = Exterior address on the program

>>> MPCC_SpeedSet : Reset broadcast speed
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11110111

>>> MPCC_SpeedUp : Speeds up the broadcast speed
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11111000

>>> MPCC_SpeedDown : Slow broadcast speed
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11111001

>>> MPCC_ToneUp : Broadcast rising tone
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11111010

>>> MPCC_ToneDown : Broadcast drop tone
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11111011

>>> MPCC_User : User from definition timbre
Square/Triangle/Noise musical part
db %11111100
dw Address
Address Is the user definition timbre data table
DPCM musical part
db %11111100,a,b,c,d

a Correspondence $4010
b Correspondence $4011
c Correspondence $4012
d Correspondence $4013

>>> MPCC_Call :Carries out exterior procedure
Square/Triangle/Noise/DPCM musical part，Generally use in Square musical part
db %11111101
dw Address
Address = Exterior address on the program

>>> MPCC_Jump : Jumps to the address which assigns continue to broadcast
Square/Triangle/Noise/DPCM musical part
db %11111110
dw Address
Address = Transfer address

>>> MPCC_End : Broadcast data conclusion
Square/Triangle/Noise/DPCM musical part
db %11111111

==========

Timbre control data form:MTCC....Music Tone Control Code

==========

>>> MTCC_wait : Waiting NMI
db %0-cccccc
c = Waiting NMInumber(0-127)

>>> MTCC_Out : Output to PSG Buffer
db %1000abcd[,A][,B][,C][,D]
abcd = Output mask
0 = Non-correspondence data output
1 = Correspondence data output

>>> MTCC_Put : Establishment PSG Buffer
db %1001abcd[,A][,B][,C][,D]
abcd = Establishment mask
0 = Non-correspondence data
1 = Correspondence data
A = ToneOutData0
B = ToneOutData1
C = ToneOutData2
D = ToneOutData3

>>> MTCC_Or : Or PSG Buffer
db %1010abcd[,A][,B][,C][,D]
abcd = Or mask
0 = Non-correspondence data
1 = Correspondence data
A = ToneOutData0
B = ToneOutData1
C = ToneOutData2
D = ToneOutData3

>>> MTCC_Output : Establishment and output PSG Buffer

db %1011abcd[,A][,B][,C][,D]
abcd = Establishment and output mask
0 = Non-correspondence data
1 = Correspondence data
A = ToneOutData0
B = ToneOutData1
C = ToneOutData2
D = ToneOutData3

>>> MTCC_Setup : Establishment Index output
db %1100abcd,--cccccc
abcd = Output Mask，Is effective to the IndexOut series order
cccccc = Output gap value

>>> MTCC_IndexOut : Establishment index and output PSG buffer
db %1101--00, %abcdnnnn
db data1,data2....data?
abcd = Establishment mask
nnnn = Output quantity
First output waits for the gap value again.

>>> MTCC_VolumeIndexOut : Volume adds index and output PSG buffer
db %1101--01, %----nnnn
db data1,data2....data?
nnnn = Output quantity
First output waits for the gap value again.

>>>MTCC_ToneIndexOut : Note index adds and output PSG buffer
db %1101--10, %----nnnn
db data1,data2....data?
nnnn = Output quantity
First output waits for the gap again.

>>> MTCC_VolumeToneIndexOut : Volume and note index adds and output PSG buffer
db %1101--11, %----nnnn
db data1,data2....data?
nnnn = Output quantity
First output waits for the gap again.

>>> MTCC_Loop : Circulation
db %1110cccc
dw Address
cccc = Cycle-index
Address = Circulation address

>>> MTCC_VolumeInc : Volume adds one
db %11110000

>>> MTCC_VolumeDec : Volume decrease one

db %11110001

>>> MTCC_VolumeBne : Volume is not equal to data jumps the extension
db %11110010, data
dw Address
Address = Transfer address

>>> MTCC_VolumeBeq : Volume is equal to data jumps the extension
db %11110011, data
dw Address
Address = Transfer address

>>> MTCC_VolumeReset : Volume restores for the default value
db %11110100

>>> MTCC_VolumeClear : Clear the volume
db %11110101

>>> MTCC_Flag2Beq : Music_Flag2 is equal to data jumps the extension
db %11110110,data
dw Address
Address = Transfer address

>>> MTCC_Flag2Bne : Music_Flag2 is not equal to data jumps the extension
db %11110111,data
dw Address
Address = Transfer address

>>> MTCC_ToneAdd : Note adds data
db %11111000,data

>>> MTCC_SetpIndexOut : Single step index establishment and output PSG buffer，every time transfers time output time
db %11111001, %abcd----
dw Address
abcd = Establishment mask
Address = Index address
First output waits for the gap again，Index shift as Music_ToneControl4，each index automatic increase，
Music_ToneControl4 :In never use in the IndexOut series order situation is zero.

>>> MTCC_Flag1Beq : Music_Flag1 is equal to data jumps the extension
db %11111010, data
dw Address
Address = Transfer address

>>> MTCC_Flag1Bne : Music_Flag1 is not equal to data jumps the extension
db %11111011, data
dw Address
Address = Transfer address

>>> MTCC_LoopBeq : If the circulation decrease counter is equal to data jumps the extension

db %11111100,data
dw Address
Address = Transfer address

>>> MTCC_Call : Carries out exterior procedure
db %11111101
dw Address
Address = Exterior address on the program
In don't use the IndexOut series order under the premise，exterior procedure may use the register has：
Music_ToneControl2 High four bytes
Music_ToneControl3 All
Music_ToneControl4 All
Music_Flag1 All
Music_Flag2 All

>>> MTCC_Jump Unconditional jump
db %11111110
dw Address
Address = Transfer address

>>> MTCC_End : Timbre control data sheet conclusion
db %11111111
