MAmidiMEmo
The Virtual S/W Synthesizer

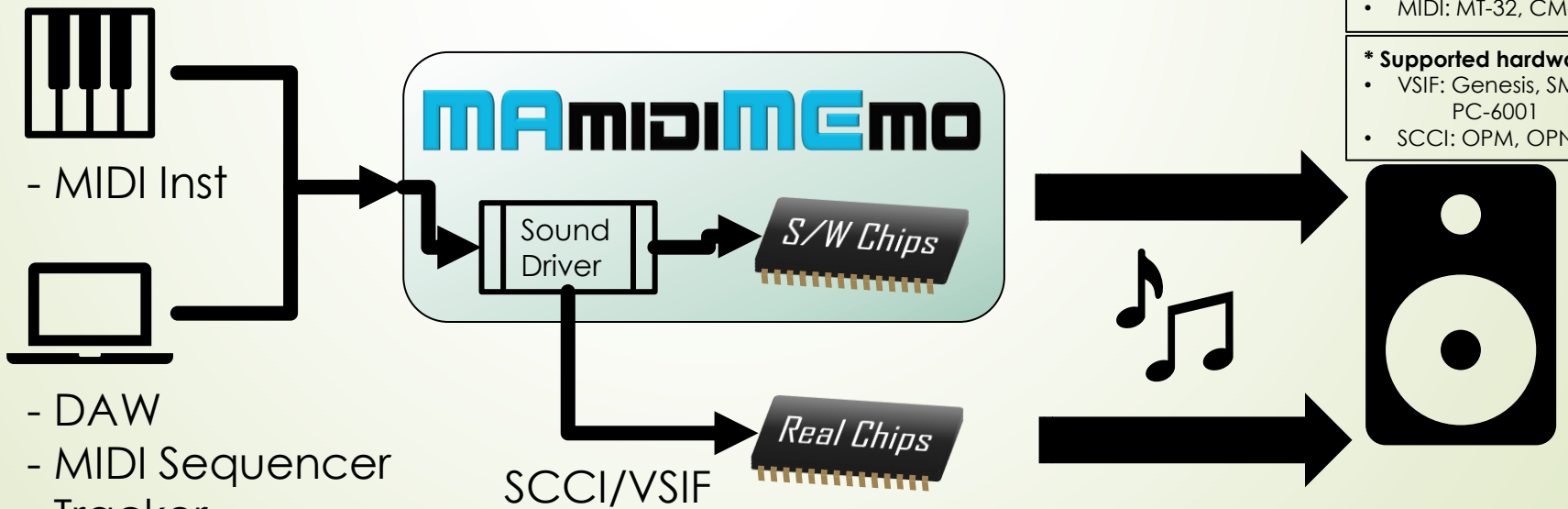
User's Manual – for MAmidiMEmo V4.5.8.0

What is the MAmidiMEmo?

- MAmidiMEmo is a virtual chiptune sound MIDI module for Windows
- You can use MIDI or DAW to sound the MAmi
- MAmi supports various sound chips*
- Also, MAmi can drive real hardware chips* via SCCI, VSIF

*** Supported chips are the following**
- PCM: C140, SPC700
- FM Synthesis: OPM, OPN2, OPNA, OPLL, OPL, OPL3
- WSG: NAMCO CUS30, HuC6280, SCC
- PSG: SID, POKEY, GB APU, SN76496, NES APU, MSM5232, AY-3-8910
- VOICE: TMS5220, SP0256, SAM
- MIDI: MT-32, CM-32P(Simulation)

*** Supported hardware is the following**
- VSIF: Genesis, SMS, Famicom, MSX, C64, PC-6001
- SCCI: OPM, OPNA, OPZ



Install & Basic Settings

- Install & run
  - Extract the downloaded zip file.
  - Click MAmidiMEmo.exe and please press [Allow access] button. This is because MAmidiMEmo uses interprocess communication technology to communicate with other apps in order to sound from the app.

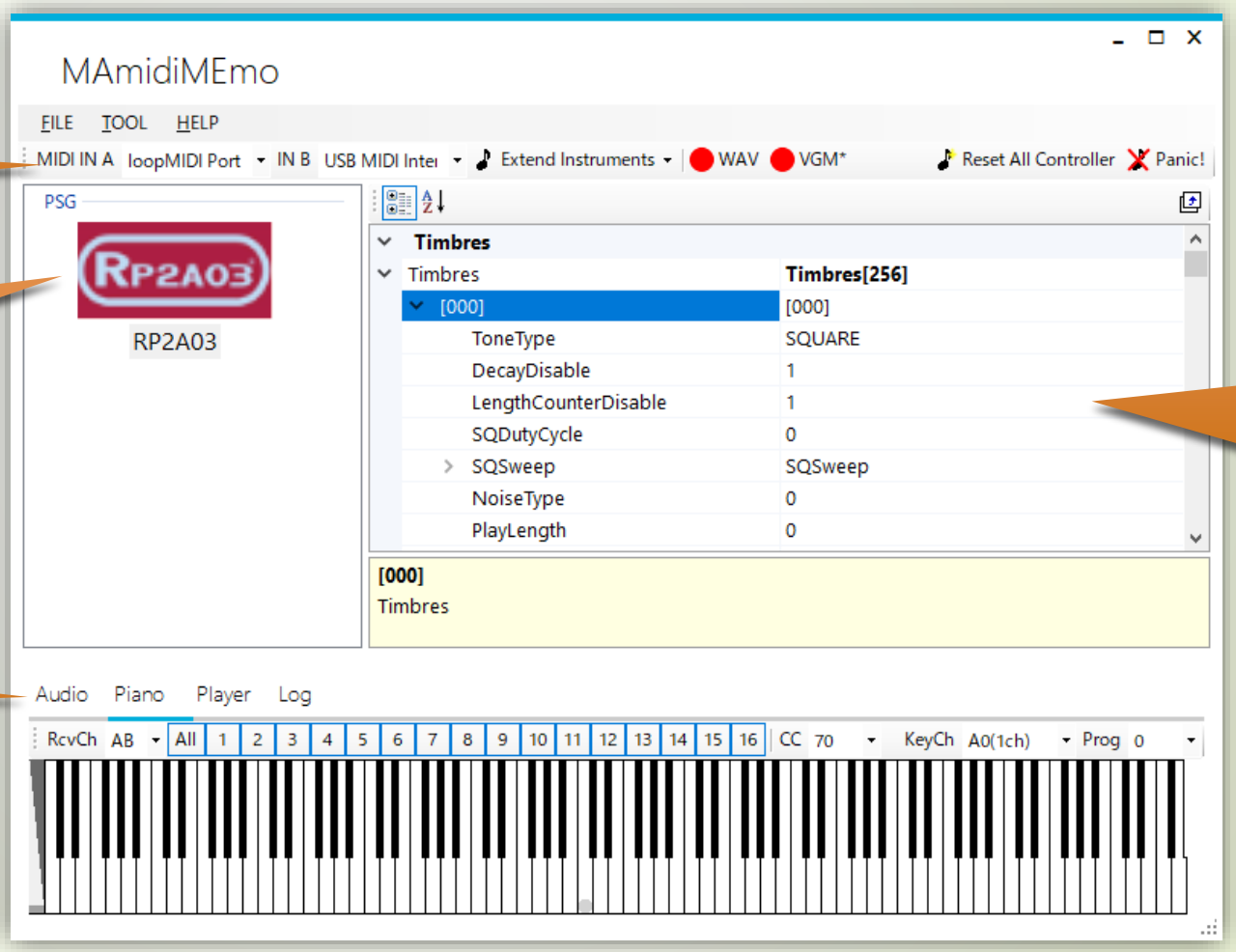
  - Will open the MAmidiMEmo. If not, please check the followings.
    - **.NET Framework 4.7 or later** installed on your PC.
    - **VC++ 2012 Runtime** installed on your PC.
    - (Execute "DelZoneID.ps1“ to remove “Zone.Identifier” flag.)

Window Overview

MIDI IN A,B Selector

Active Chips (see next)

Tools

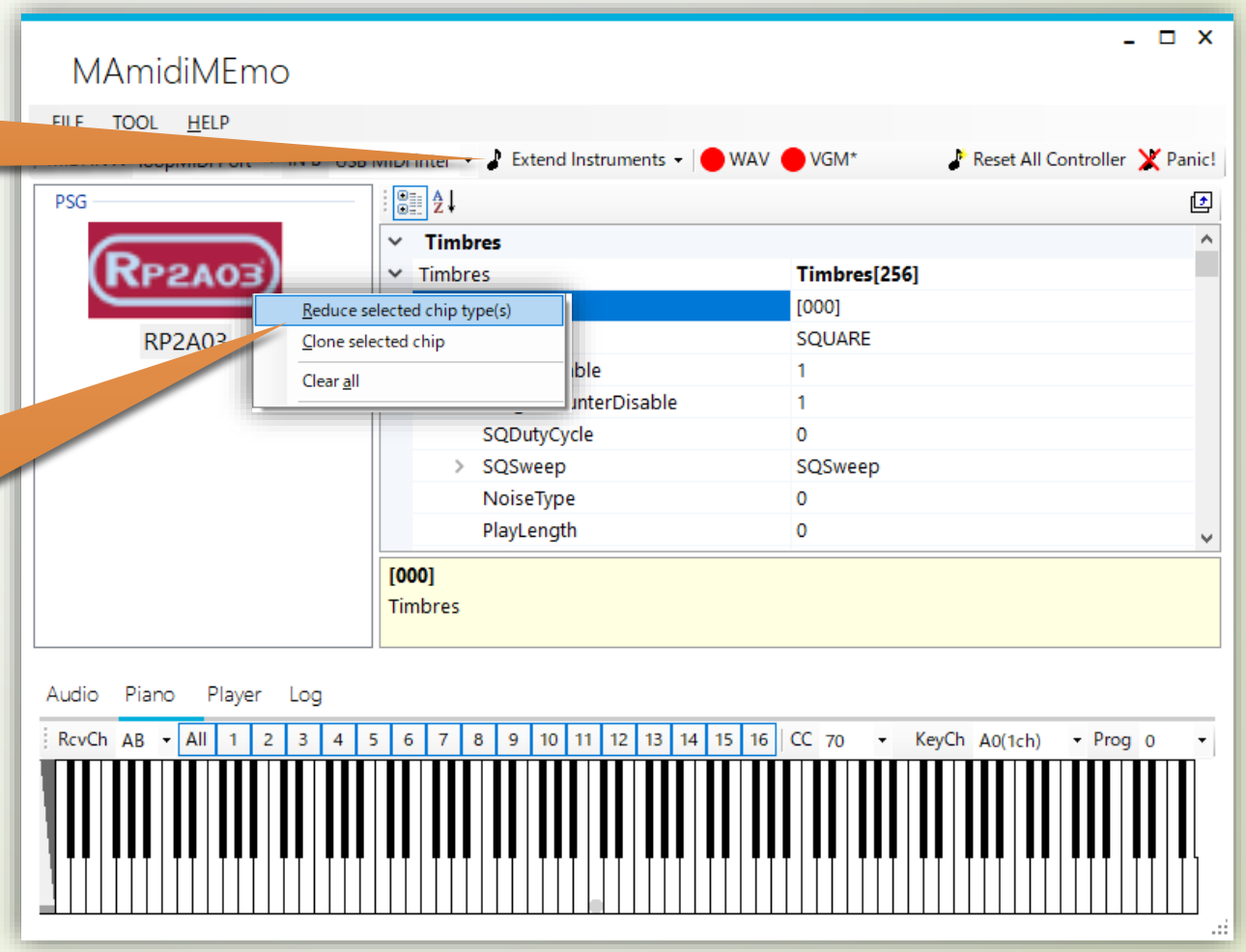


Chip Parameter Editor (see next)

Add and Remove a Chip

To add
Select the chip from this menu.

To remove
Open a context menu and select.

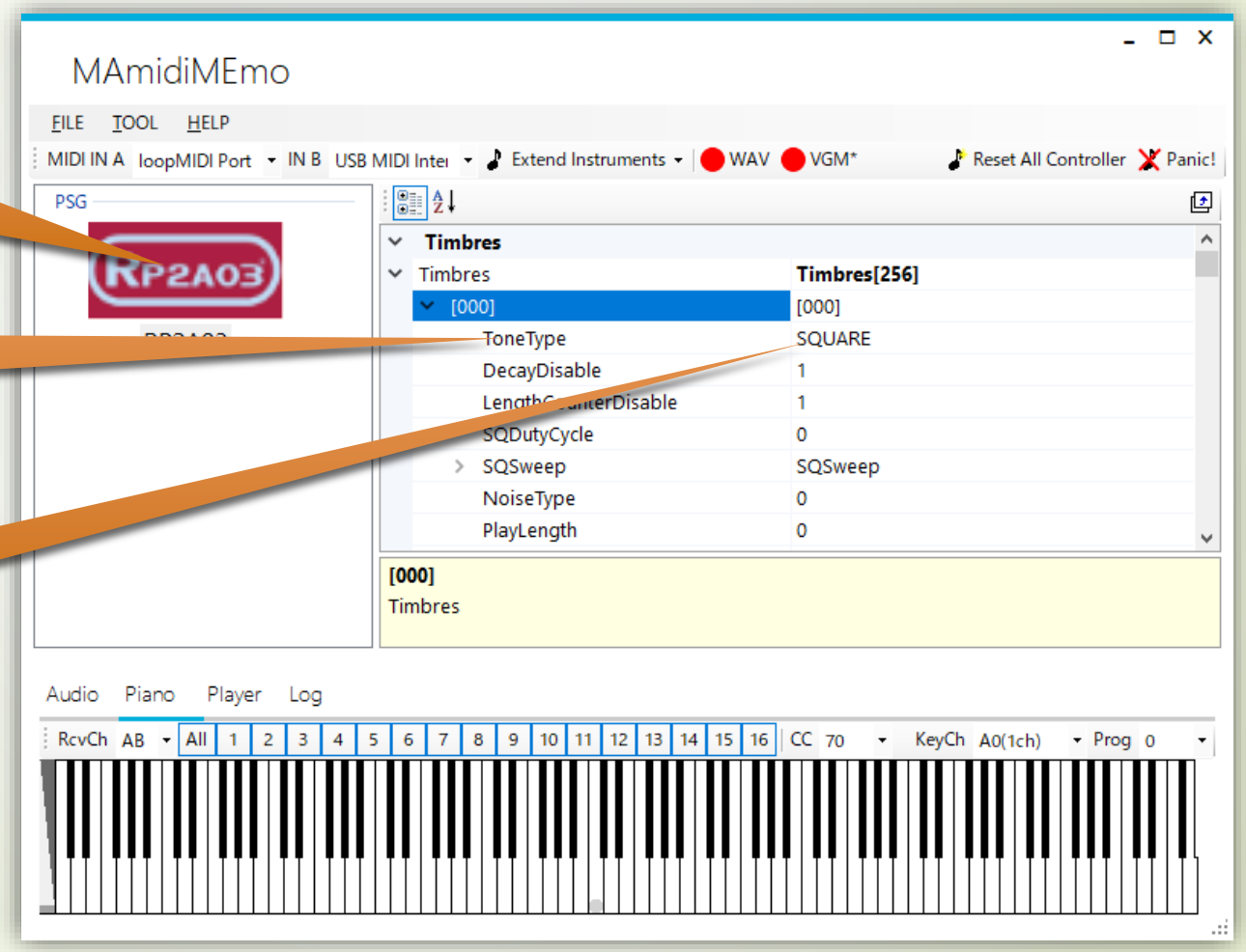


Edit chip and sound parameters

1. Click chip

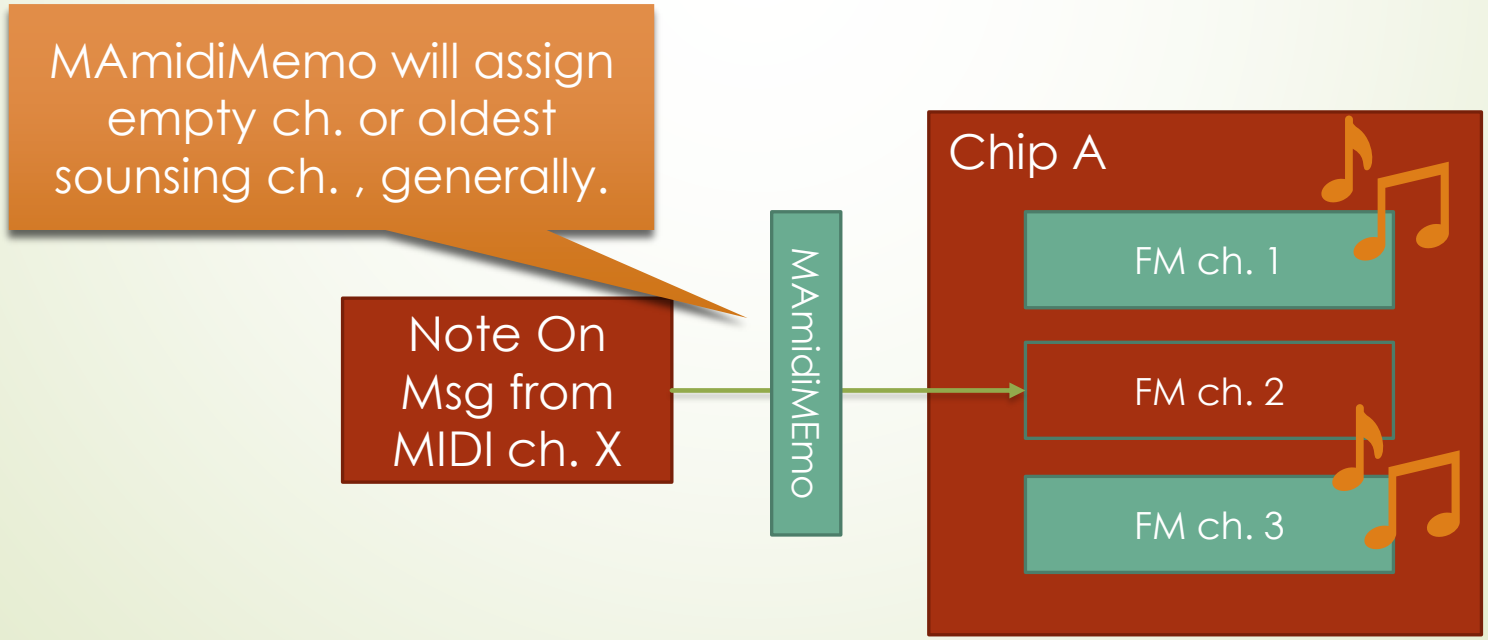
2. Click parameter

3. Change value



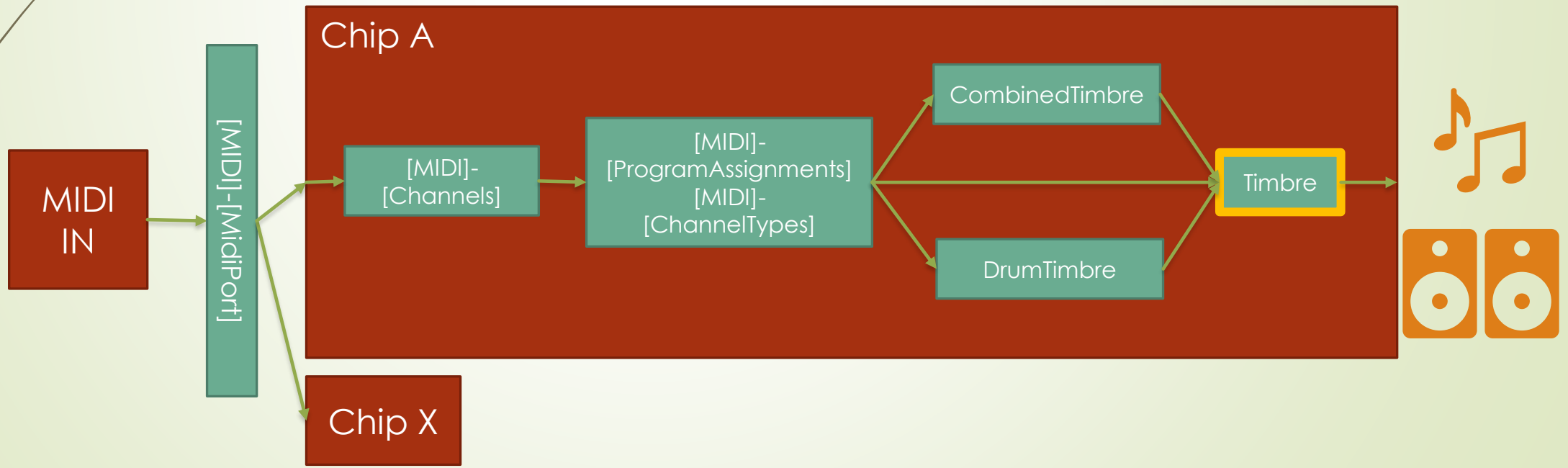
Between MIDI ch and Chip ch Relation.

- You don't need to concern the Chip ch. , generally.
  MAmidiMEmo will assign suitable Chip ch. automatically.
  However, you need to concern a max ch. number of the Chip.
- MAmidiMEmo will assign oldest sounding ch. to sound the new sounds.



Sounding Structure

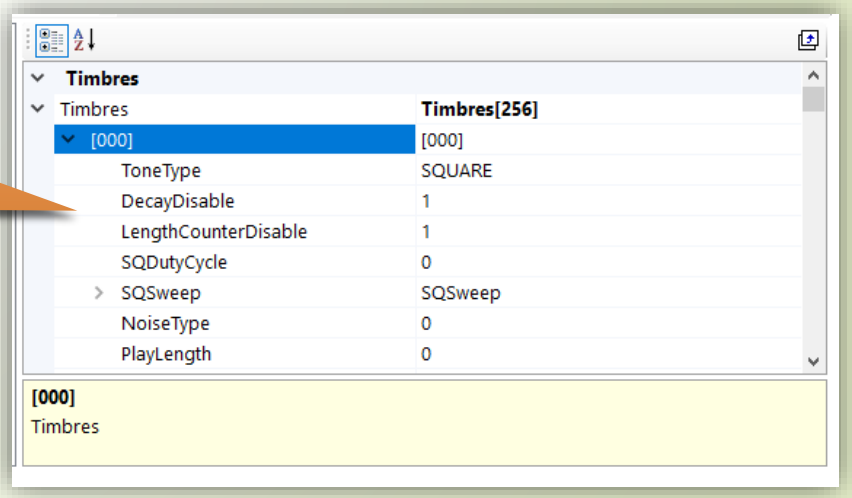
- MAmidiMEmo outputs a sound from MIDI message along with the following structure.
  So, at least, you need to edit the **Timbre** parameters to sound something.



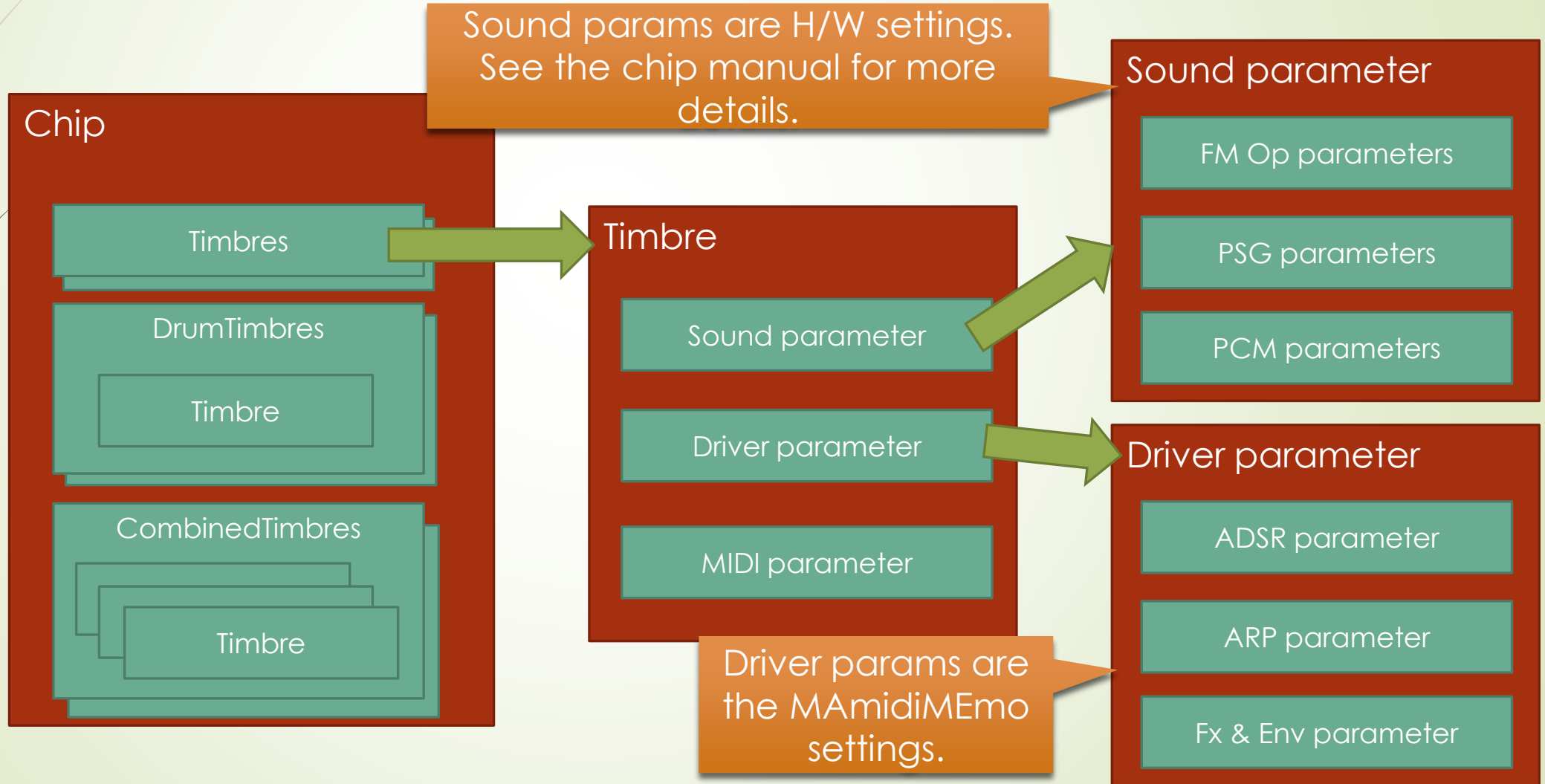
Timbre

- Generally, a chip has 256 Timbres, 256 CombinedTimbres, 128 DrumTimbres.
- CombinedTimbre can sound multiple Timbers at the same time (up to 4)
- DrumTimbre can sound Timbes as a Drum sounds (Ignoring Note Off msg).
- You can change the Timbre parameters on the Chip Parameter Editor. Generally, you need to learn the chip specification to edit the chip parameters.

Chip Parameter Editor

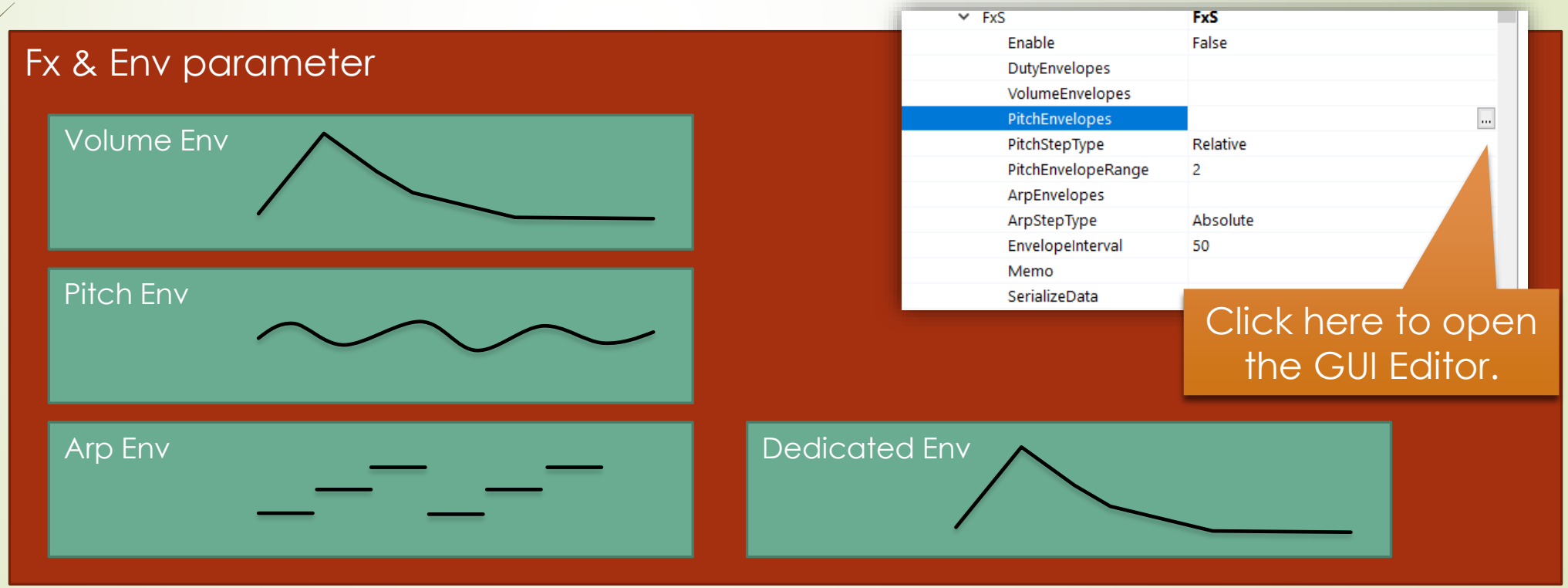


Timbre Structure



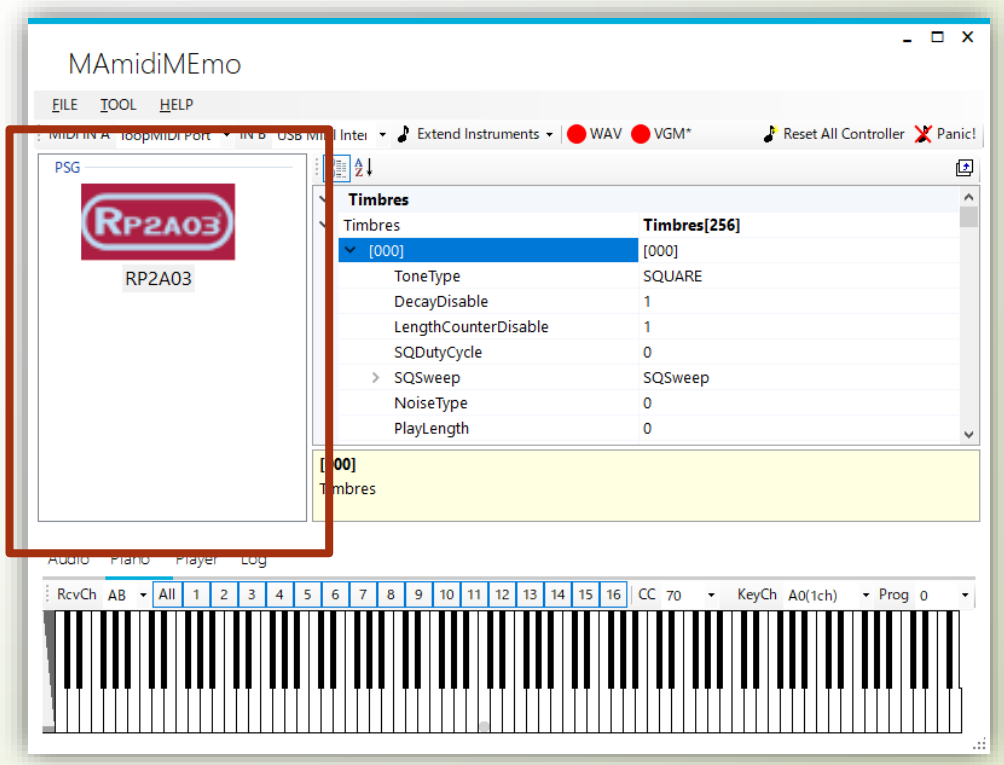
Driver parameters - Fx & Env Structure

- You can make for a rich sound by using driver params. Especially, FxS can do it.



Sample sounds

- There are sample sound files in the “Samples” folder. You can drop a sample file “*.MAmi" to the left pane.



Additional files

- YM2608
  - Place legitimate “ym2608_adpcm_rom.bin” file in the MAmidiMEmo directory to sound ADPCM rhythm sounds.
- MT-32
  - Place legitimate “MT32_CONTROL.ROM” and “MT32_PCM.ROM” in the MAmidiMEmo directory to sound ADPCM sounds.

Limit Break

- Any chip can output only a few voices. However, MAmidiMEmo can break this limitation by the following steps.
  1. Add a chip and complete all settings the chip.
  2. Select the [Clone selected chip] Cloned chip added.
  3. Select the cloned chip and set the [Follower Mode] value to “Unit0*”.
     * If clone source chip ID is 0.

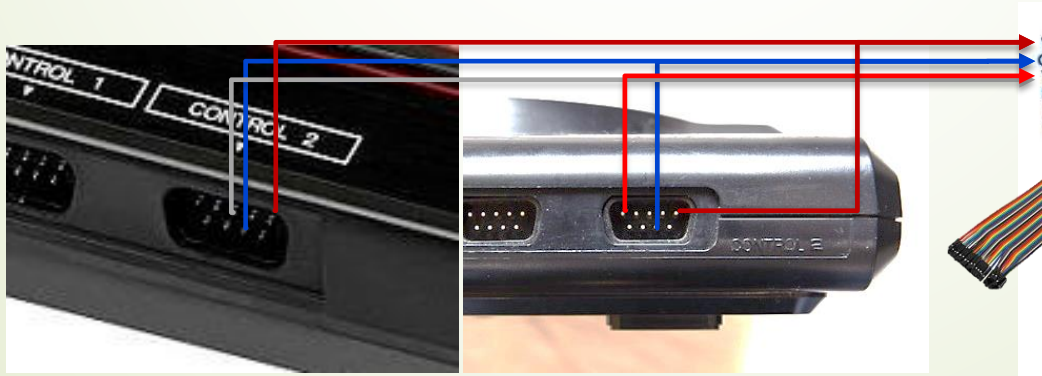
- When the clone source chip consumed all voices, the cloned chip sound for the chip.
- If you want to extend max voices more, select the [Clone selected chip] of the cloned chip. And set the [Follower Mode] value to “Unit0”.

Use a real hardware

You can use a real hardware by using the VSIF / PCI device.

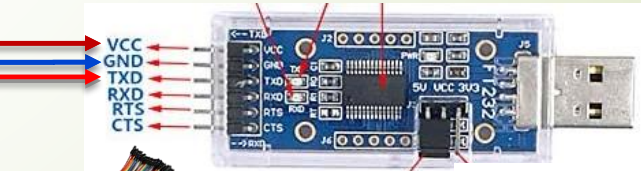
VGM Sound Interface(VSIF - UART) for Genesis/SMS

- MAmidiMEmo and VGMPlayer can drive real machine chips. Currently supports NTSC SMS(2, *Mk*Ⅲ) for SN76489, OPLL and NTSC Genesis(MD) for SN76489, OPN2.
- How to
  1. Buy the following parts.
     - 1x UART dongle (Note: FT232R and so on. CH340 and CP2102 may not work 163,840bps, only 115,200bps.)
     - 1x FLASH Cart for SMS or Genesis and 1x D-SUB 9 pin female connector and DuPont wires
  2. Solder like the following and connect it to the JOYSTICK PORT 2.



SMS/2 or *Mk*Ⅲ

Genesis/MD



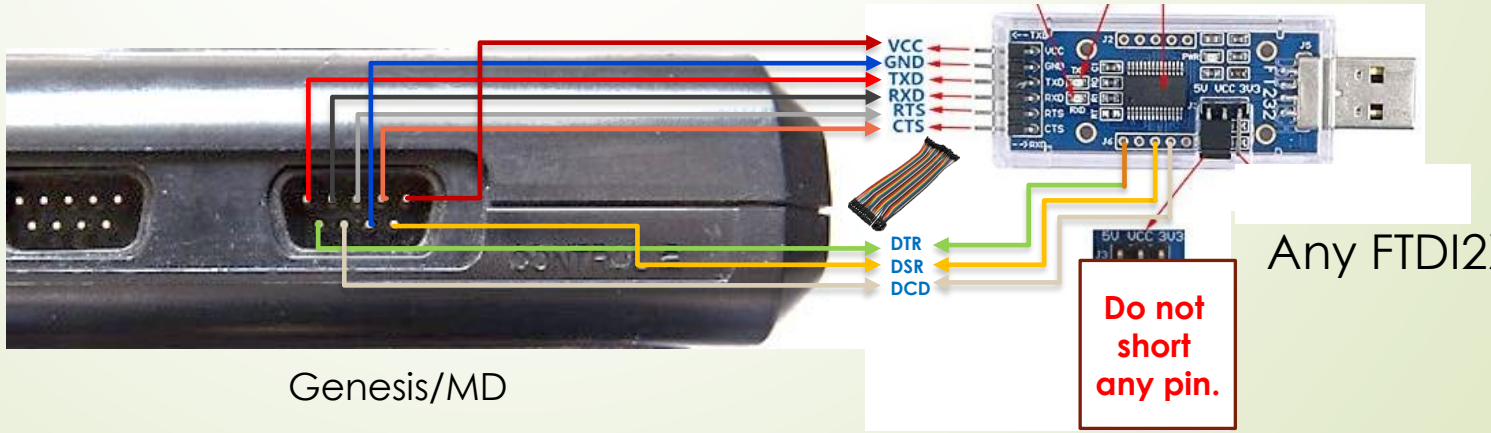
Any UART dongle

Do not short any pin.

*Need to support
163,840bps for MD
115,200bps for MD, SMS

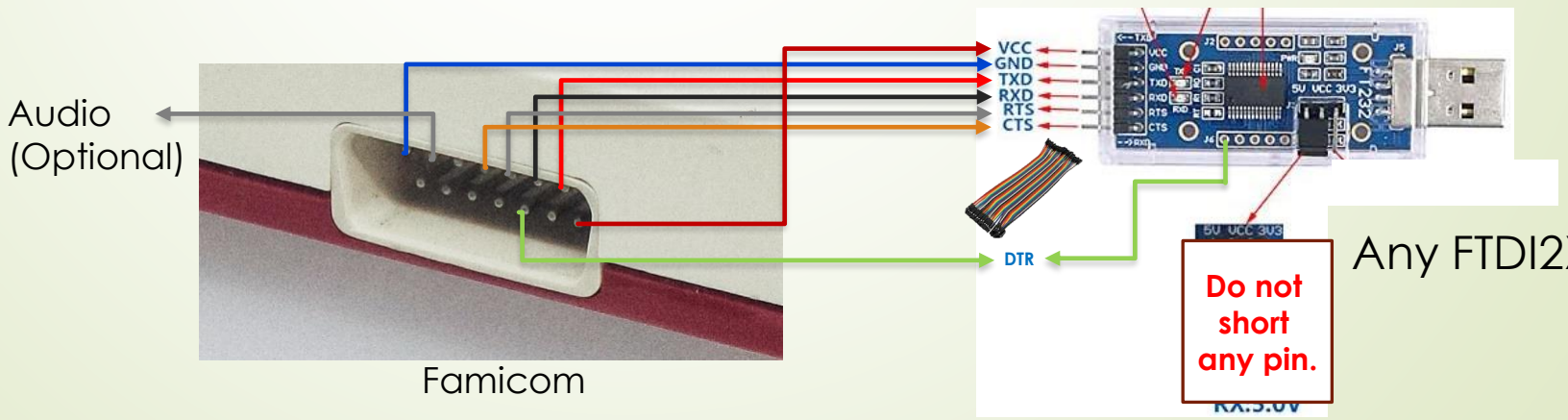
VGM Sound Interface(VSIF - FTDI) for Genesis

- MAmidiMEmo and VGMPlayer can drive real machine chips more faster if you use FTDI2xx(232R, 232H and so on). Currently supports NTSC Genesis(MD) for SN76489, OPN2.
- How to
  1. Buy the following parts.
     - 1x FTDI2XX dongle (FT232R or FT2232H and so on. Need to support 5V. ***WARN* Please use genuine dongle.**)
     - 1x FLASH Cart for Genesis and 1x D-SUB 9 pin female connector and DuPont wires
  2. Solder like the following and connect it to the JOYSTICK PORT 2.



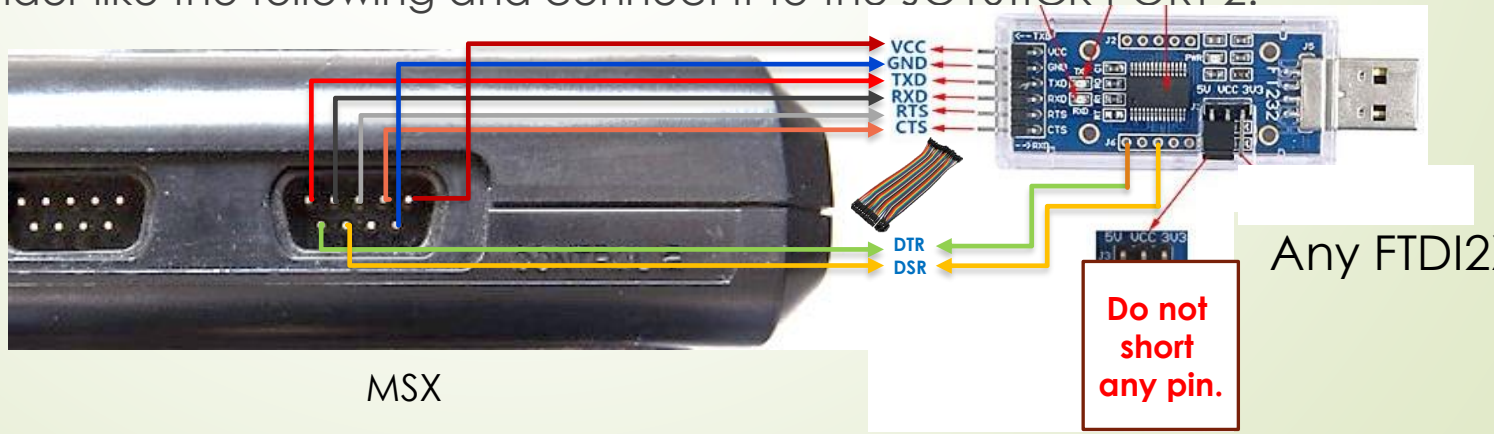
VGM Sound Interface(VSIF - FTDI) for Famicom

- MAmidiMEmo can drive real machine chips more faster if you use FTDI2xx(232R, 232H and so on). Currently supports NTSC Famicom and RP2A03(No DAC)/FDS/VRC6.
- How to
  1. Buy the following parts.
     - 1x FTDI2XX dongle (FT232R and so on. Need to support 5V. ***WARN* Please use genuine dongle.**)
     - 1x FLASH Cart for Famicom and 1x D-SUB 15 pin female connector **for FC** and DuPont wires
  2. Solder like the following.



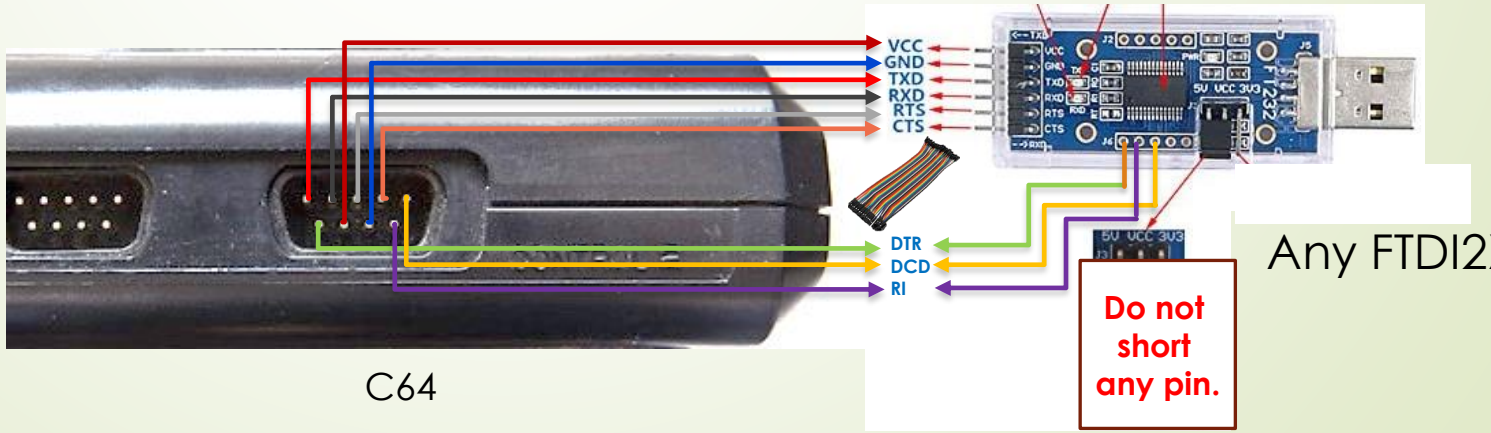
VGM Sound Interface(VSIF - FTDI) for MSX

- MAmidiMEmo can drive real MSX machine chips if you use FTDI2xx(232R, 232H and so on). Currently supports NTSC MSX for AY-3-8910 and OPLL and SCC+ and OPL3.
  NOTE: Be sure to select proper SLOT# for SCC to use SCC.
  Set FTDI clk value to 17~ for each chip.
- How to
  1. Buy the following parts.
     - 1x FTDI2XX dongle (FT232R and so on. Need to support 5V. ***WARN* Please use genuine dongle.**)
     - 1x D-SUB 9 pin female connector and DuPont wires
  2. Solder like the following and connect it to the JOYSTICK PORT 2.



VGM Sound Interface(VSIF - FTDI) for Commodore 64(C64)

- MAmidiMEmo can drive real MSX machine chips if you use FTDI2xx(232R, 232H and so on). Currently supports NTSC/PAL C64 for SIDs.
  We recommend to use ARMSID with ADSR bud fixing.
- How to
  1. Buy the following parts.
     - 1x FTDI2XX dongle (FT232R and so on. Need to support 5V. ***WARN* Please use genuine dongle.**)
     - 1x D-SUB 9 pin female connector and DuPont wires
  2. Solder like the following and connect it to the JOYSTICK PORT 2.

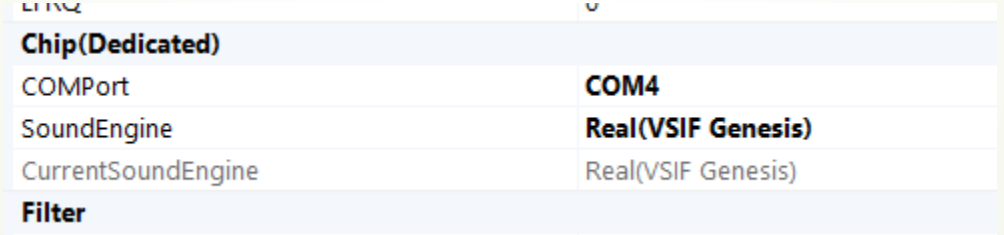


VGM Sound Interface(VSIF) Settings

1. Burn VGMPlay_md.bin(for Genesis) or VGMPlay_sms.sms(for SMS) or VGMPlay_nes*.* (for Famicom) or VGM_msx/P6.rom(for MSX*/PC-6001) or VGMPlay_c64.prg to your FLASH Cart and so on.

   *VGMPlay_nes_vrc6/fds/mmc5 ROM does not show any screen but same UI with VGMPlay_nes.nes UI
   *VGM_msx_Vkey.rom can skip booting from this ROM while the [V] key is **NOT** pressed at boot time.

2. Set the COMPort/FTDI ID and select "VSIF ***" you wish.



3. Reset your console and push [Panic!] button
4. (Famicom only)Re-send DPCM data.
5. Done!
6. If you can not sound sounds, make sure soldering and COMPort name.
   Or, please contact me.

*Some UART dongles may not work properly.
*Compatible consoles may not work properly.

VGM Sound Interface(VSIF) for Famicom spec

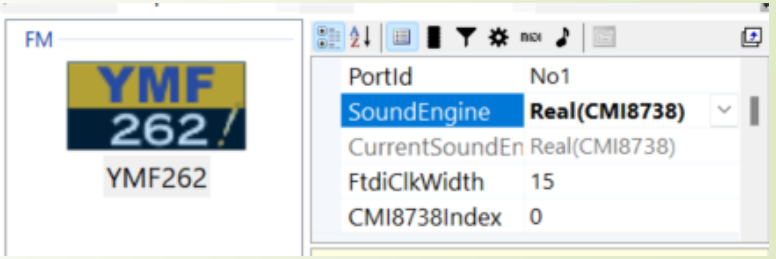
Sound	Normal ROM (Mapper 0)	FDS IMAGE*2	VRC6 ROM*2 (Mapper 24)	MMC5 ROM*2*3 (Mapper 5)
Square	OK	OK	OK	OK
Tri	OK	OK	OK	OK
Noise	OK	OK	OK	OK
DPCM	NO	OK(Up to 8KB)	NO	OK*1 (Up to 64KB)
Ext. Snd FDS	NO	OK*1	NO	NO
Ext. Snd VRC	NO	NO	OK*1	NO
Ext. Snd MMC	NO	NO	NO	NO

*1 Not Tested
*2 China flash cart may not work properly
*3 PRG-RAM 32x2 KB

Use CMI8738(OPL3) PCI Board

Please use at your own risk

1. Attach the CMI8738 Board to your PC. *Only for 64bit Windows*
2. **Disable Driver Signature enforcement**
3. (*Uninstall and remove* old CMI8738 OPL3 driver if installed.)
4. Install the CMI8738 OPL3 driver located in ".¥CMI8738OPL3" folder.
5. Set [SoundEngine] prop to the "Real(CMI8738)".
6. Have fun!!



Technical information

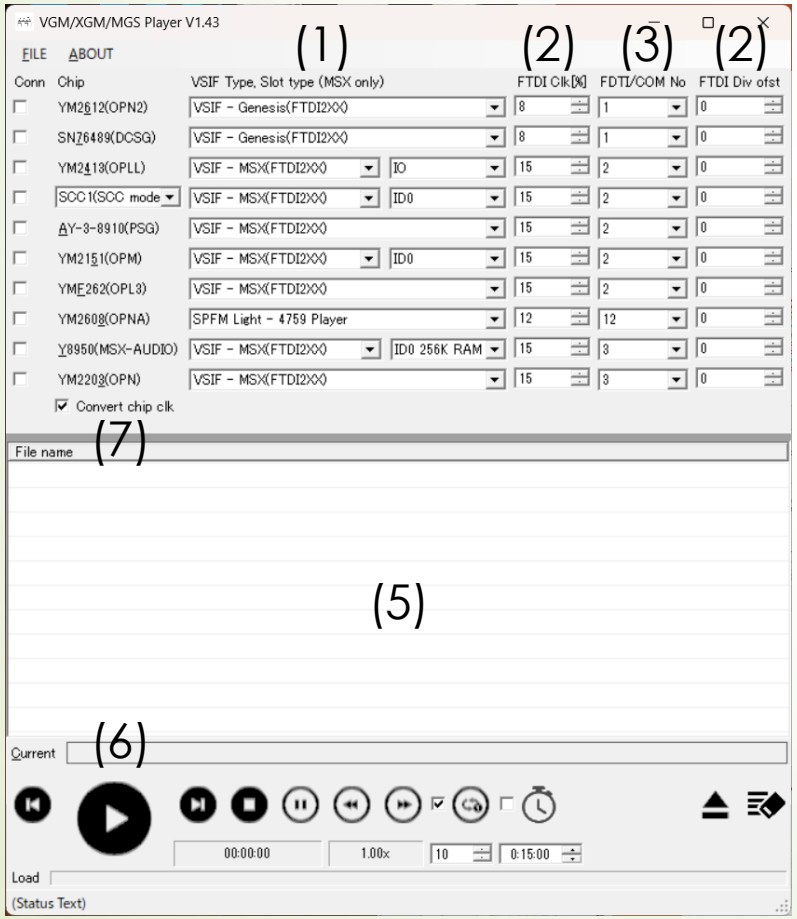
If you want to use the OPL3 of the CMI8738 directly from your app…

1. Use the helper DLL "CMI8738OPL3Library.dll".
2. Or, direct access I/O port with admin rights.
   eg) DF00H+50H is the OPL3(CMI8738) port.

VGMPlayer

VGMPlayer

VGMPlayer can play a vgm/xgm/mgs file on a real chip via VSIF or SPFM. Substitutes for similar chips are also available. For example, an OPL track can be played on an OPL3 chip.



1) Select interface type

NOTE: Bandwidth of UART is narrow. So you can not play heavy track data properly.

2) Adjust FTDI Clk for FTDIxxx mode for your environment

NOTE: Usually the default value is fine, but if the sound is strange, increase the value. If the performance is slow, decrease the value. If you can not adjust by FTDI Clk, please adjust FTDI Div ofst value.

3) Specify COMPort# for UART/SPFM:
Specify FDTI ID# for FTDI2xxx

4) Check to connect

5) Drop vgm/XGM/MGS files to here

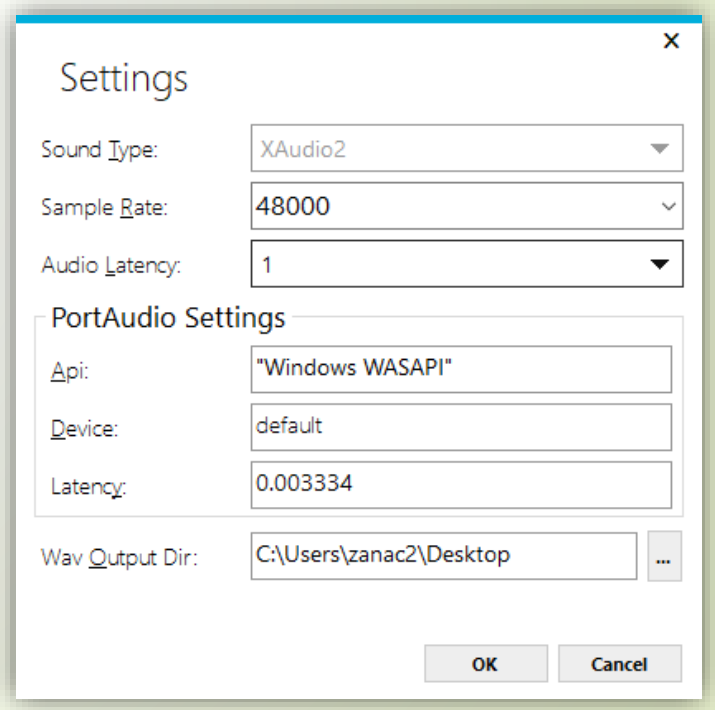
6) Push to play.

7) If the pitch is wrong, click here.

Appendix

Trouble Shooting for MAmi

- If you noticed “sound lag” or “stutter”, open the Settings dialog from [TOOL] menu. Check [Sound Type] and [Audio Latency] value.



MIDI Implementation Chart 1

*Depends on the chip

Function	Transmitted	Recognize	Remarks
Basic Channel	-	1-16: Defaut 1-16: Changed	
Note Number*	-	0-127	
Velocity*	-	Yes: Note ON No: Note OFF	
After Touch	-	No	
Pitch Bend*	-	Yes	**8192: Default**
Program Change	-	0-127	**0: Default**

MIDI Implementation Chart 2

*Depends on the chip

Function	Transmitted	Recognize	Remarks
Control Change 1	-	 Modulation	 **0: OFF**, 64: ON
5		Portamento Time	**0: Default**
6 38		Data Entry MSB Data Entry LSB	
7		Volume*	**127: Default**
10		Panpot*	**64: Default** 0: Left, 127: Right
11		Expression*	**127: Default**
16-19		GPCS1	Modify params
64		Hold 1	**0: OFF**, 64: ON
65		Portamento	**0: OFF**, 64: ON

MIDI Implementation Chart 3

*Depends on the chip

Function	Transmitted	Recognize	Remarks
Control Change 70-75,79	- 	 SCCS	Modify current timbre params
76 77 78	- - -	Mod. Rate Mod. Depth Mod. Delay	**64: Default** **64: Default** **64: Default**
80-83	-	GPCS2	Modify parameters
84	-	Portamento Ctrl	**0: OFF**, 64: ON
91-95	-	VST Plugin Ctrl	Modify VST params
98 99	- -	NRPN LSB NRPN MSB	
100 101	- -	RPN LSB RPN MSB	

MIDI Implementation Chart 4

*Depends on the chip

Function	Transmitted	Recognize	Remarks
Control Change 121	- 	 Reset All Ctrl	
126	-	Mono Mode	**0: OFF** 1-127: Max Voice Num.*
127	-	Poly Mode	**0: OFF** 1-127: Reserve Voice Num.* *Reset Mono Mode when set

MIDI Implementation Chart 5

Function	MSB	LSB	Remarks
RPN	0	0	Pitch Bend Range 0-**2**-127 [Half Note]
	0	5	Mod Depth **0**-127 [Relative]

MIDI Implementation Chart 6

Function	MSB	LSB	Remarks
NRPN	0	16-19 80-83	GPCS[1-4] Value GPCS[5-6] Value 0-127
	0	70-75 79	SCCS[1-6] Value SCCS[10] Value 0-127

MIDI Implementation Chart 7

Function	Change Receiving MIDI ch. dynamically.	Remarks
NRPN	NRPN MSB Bx 63 41 ⋯ for MIDI ch(1-7) NRPN LSB Bx 62 <Device ID> ... Specify Device ID of existing instrument. DATA MSB Bx 26 <Unit No> ... Specify Unit No of the above Device ID of existing instrument. DATA LSB Bx 06 <Receiving MIDI ch(1-7) bit sets. 1=On, 0=Off> bit 6 5 4 3 2 1 0 ch 7 6 5 4 3 2 1 NRPN MSB Bx 63 42 ... for MIDI ch(8-14) NRPN LSB Bx 62 <Device ID> ... Specify Device ID of existing instrument. DATA MSB Bx 26 <Unit No> ... Specify Unit No of the above Device ID of existing instrument. DATA LSB Bx 06 <Receiving MIDI ch(8-14) bit sets. 1=On, 0=Off> bit 6 5 4 3 2 1 0 ch 14 13 12 11 10 9 8 NRPN MSB Bx 63 43 ... for MIDI ch(15-16) NRPN LSB Bx 62 <Device ID> ... Specify Device ID of existing instrument. DATA MSB Bx 26 <Unit No> ... Specify Unit No of the above Device ID of existing instrument. DATA LSB Bx 06 <Receiving MIDI ch(15-16) bit sets. 1=On, 0=Off> bit 6 5 4 3 2 1 0 ch xx xx xx xx xx 16 15	

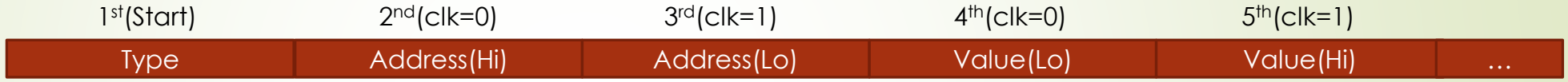
VSIF – Generic (UART 115K), SMS(UART 115K) SPECIFICATION for AY-3-8910, YM2413

- Baud rate: 115,200 bps
- Protocol: 8 bits, None parity bit, 1 stop bit
- 1 packet : 2 bytes

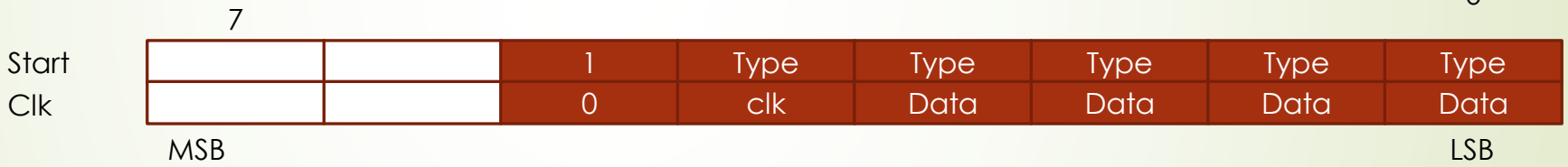
VSIF – MSX/P6(FTDI) SPECIFICATION

for AY-3-8910, OPLL, SCC-I, OPL3, OPM, OPNA/OPN2, OPN

- Baud rate : 38,400 bytes / sec
- 1 packet : 5 bytes + a



- 1byte : 4bit(data) + 2bit(Start + Clk bit)

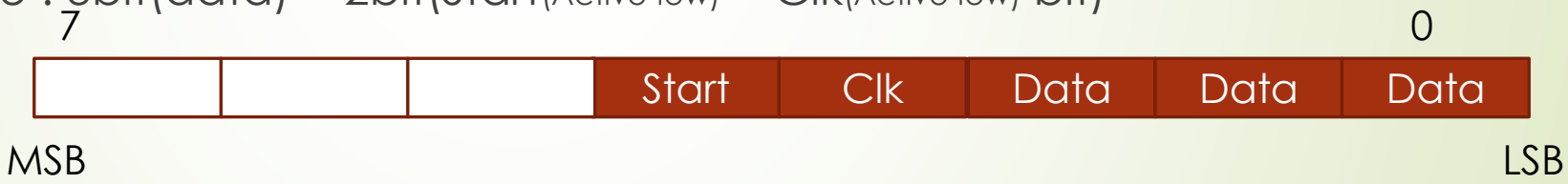


- Type:
  - **0** AY-3-8910: Write value to address
  - **1,2,(12)** YM2413: **1** is write value to address, **2** is set OPLL cartridge slot number
  - **3∼9** SCC-I: *(in preparation)*
  - **10∼11** YMF262: **10** is write value to address of port L , **11** is Write value to address of port H
  - **(13),14** OPM: Write value to address
  - **15** DCSG: Write value to address
  - **16 ∼17** OPNA/OPN2: Write value to address
  - **18** OPN: Write value to address

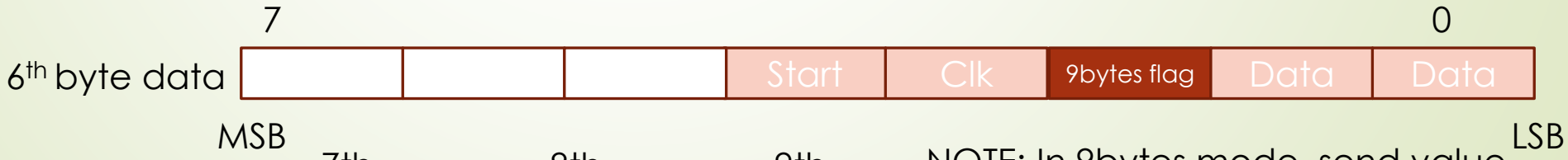
VSIF – C64(FTDI) SPECIFICATION for SID

- Baud rate : 31,250 bytes / sec
- 1 packet : 6 bytes(1 byte value) or 9 bytes (2 bytes value) mode

- 1byte : 3bit(data) + 2bit(Start(Active low) + Clk(Active low) bit)



- 9 bytes mode flag:



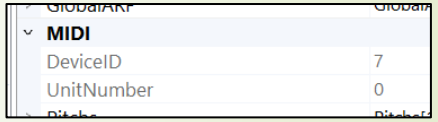
NOTE: In 9bytes mode, send value for Address+1 data first, second is Address+0 data

Integrate with MAmidiMEmo via RPC

- MAmidiMEmo will starts RPC server on port 30000 at startup.
  So, your application can be integrate with the MAmidiMEmo via RPC.
  MAmidiMEmo provides the following API

```
//Write the value to the address to the specific chip
void DirectAccessToChip(unsigned char DeviceID, unsigned char UnitNo, unsigned int address, unsigned int data)
```

* You can confirm the DeviceID and UnitNumber from the property.
* Currently, OPLL(ID9), SCC(ID7), AY8910(ID11) chips are supported.
* If you want to use SCC**1**(aka SCC**+**), you need to add 0x100 to the address.



- Ex)

```
rpc::client* m_rpcClient = new rpc::client("localhost", 30000);   //Open RPC port
:::
//Write SCC1 wave form
m_rpcClient->async_call("DirectAccessToChip", (unsigned char)7, (unsigned char)0, (unsigned int)0x100, (unsigned int)10);
//All sound off for AY8910
m_rpcClient->async_call("DirectAccessToChip", (unsigned char)11, (unsigned char)0, (unsigned int)7, (unsigned int)0x3f);
//Key off 1ch for OPLL
m_rpcClient->async_call("DirectAccessToChip", (unsigned char)9, (unsigned char)0, (unsigned int)0x20, (unsigned int)0x0);
:::
m_rpcClient->~client(); //Close RPC port
```