ŠABLONA VST PLUG-IN MODULU

Dokument verze 1.2, 4. 5. 2008, Jiří Schimmel, Vysoké učení technické v Brně

Nastavení projektu

Projekt je určen pro vývojové prostředí MS Visual C++ s verzí projektu 8 (MS Visual C++ 2005 a vyšší). Kromě standardní konfigurace pro ladění *Debug* a *Release*, kdy je plug-in modul sestaven v podadresáři `Debug` resp. `Release`, je k dispozici identická konfigurace *Debug with GTPlayer EDU*, která sestavuje plug-in modul v adresáři plug-in modulů tohoto programu. Při první kompilaci budete prostředím Visual C++ vyzváni k uložení tzv. *Solution*, což je sestava projektů otevřená v programu (v případě šablony VST plug-in modulu je to jediný projekt).

Soubory určené pro editaci uživatelem

`user_variables.h`

pomocné proměnné potřebné pro implementaci algoritmu. V šabloně je pomocí direktivy `include` zařazen do bloku `protected` definice třídy VST plug-in modulu. Uvnitř souboru `user_variables.h` lze samozřejmě použít specifikátory `public` a `private`, pokud je potřeba použít jiná přístupová práva.

`vst_temp_defs.h`

1) definice maker a výčtových typů určujících základní vlastnosti plug-in modulu:

- `def_canDoubleReplacing`: pokud je definováno, je podporována funkce `processDoubleReplacing()`
- `def_Editor`: pokud je definováno, je použit editor
- `def_tempInput`: pokud je definováno, nepracuje funkce `processReplacing` přímo se vstupními daty předanými z hostitelské aplikace, ale s jejich kopií.
- `def_tempOutput`: pokud je definováno, neukládá funkce `processReplacing` výstupní data přímo do výstupní vyrovnávací paměti hostitelské aplikace, ale do dočasné.
- `def_VendorVersion`: definice verze (číslo datového typu 32bit integer)
- `def_UniqueID`: definice jednoznačného identifikátoru (4 znakový řetězec)
- `def_isSynth`: pokud je definováno, plug-in modul se chová jako VST instrument (podpora v šabloně není kompletní).
- `def_ChNumber`: počet vstupních a výstupních kanálů
- `def_ProgNumber`: počet programů (ve skutečnosti pouze jeden, podpora více programů není implementována).
- `def_RowsInEditor`: počet řádků parametrů v editoru; pokud není definováno, jsou všechny parametry v editoru v jednom sloupci

2) definice pojmenovaných konstant indexů parametrů pomocí výčtového typu `tagTemplateParameter`

```
enum tagTemplateParameter {
   kGain = 0,
   kBypass,
   PARAM_NUMBER,
};
```

- číslování konstant ve výčtovém typu musí začínat od nuly
- poslední řádek ve výčtovém typu musí být `PARAM_NUMBER`
- pro přemostění efektu je předdefinována konstanta `kBypass`

`vst_temp.cpp`

pro editaci uživatelem je určena pouze část kódu vymezená poznámkami `USER CODE` a `END OF USER CODE`.

1) definice konstant

- `char def_DefaultProgName[kVstMaxProgNameLen]`: standardní jméno programu
- `char def_VendorString[kVstMaxVendorStrLen]`: textový identifikátor výrobce
- `char def_ProductString[kVstMaxProductStrLen]`: textový identifikátor produktu
- `char def_EffectName[kVstMaxEffectNameLen]`: jméno efektu
- uživatelem definované konstanty pro algoritmus, např. Ludolfovo číslo.

2) pro definici reálných hodnot, vlastností a chování parametrů algoritmu je určeno pole struktur `unsortedParams`:

```
ParamInfo params[PARAM_NUMBER] = {
      ParamInfo param1,
      ParamInfo param2,
      ...
      ParamInfo lastparam };
```

kde strukturu ParamInfo lze přímo inicializovat

```
      { int tag,
        0.0f,
        float default,
        float min,
        float max,
        char name[],
        char label[],
        char type,
        bool readonly }
```

- `tag`: pojmenovaná konstanta indexu odpovídajícího parametru definovaná ve výčtovém typu `tagTemplateParameter` v souboru `vst_temp_defs.h`
- `default`: standardní skutečná hodnota parametru, která se nastaví po otevření modulu
- `min`: minimální skutečná hodnota parametru
- `max`: maximální skutečná hodnota parametru
- `name`: jméno parametru (maximálně 8 znaků)
- `label`: označení parametru (jednotky, maximálně 8 znaků)
- `type`: typ zobrazení parametru
  - `SHOW_VOID`: nedefinované zobrazení
  - `SHOW_FLOAT`: zobrazení jako číslo s plovoucí řádovou čárkou
  - `SHOW_LOG`: zobrazení jako číslo s plovoucí řádovou čárkou, logaritmická stupnice
  - `SHOW_IN_DB`: číslo typu `float` je automaticky přepočteno na dBFS
  - `SHOW_IN_HZ`: číslo udávající počet vzorků na periodu signálu převede na Hz podle aktuálního vzorkovacího kmitočtu
  - `SHOW_IN_MS`: číslo udávající počet vzorků převede na ms podle aktuálního vzorkovacího kmitočtu
  - `SHOW_IN_PC`: číslo typu `float` zobrazí v procentech (bez desetinných míst)
  - `SHOW_INTEG`: určeno pro přepínače nabývající pouze kladných celých hodnot
  - `SHOW_ONOFF`: určeno pro dvoupolohové přepínače zapnuto/vypnuto

- `readonly`: pokud je `true`, je parametr určen pouze pro čtení
  Při definování pole `unsortedParams` není nutné dodržet pořadí parametrů, jak byly pojmenované konstanty jejich indexů definovány ve výčtovém typu `tagTemplateParameter` v souboru `vst_temp_defs`.h, po načtení plug-in modulu jsou parametry automaticky setříděny podle indexů výčtového typu.

3) Funkce pro inicializaci a deinicializaci uživatelských proměnných definovaných v souboru `user_variables.h`.
   - `setUserVariables`: inicializace statických proměnných a alokování dynamických proměnných
   - `clearUserVariables`: dealokování dynamických proměnných
4) Funkce pro práci s parametry
   - `parameterChanged`: tato funkce je volána při každé změně parametru, pokud vrátí `true`, je změna povolena, pokud vrátí `false`, povolena není. Tuto funkci není nutné implementovat, je ale vhodná v případě, kdy je nutné převádět parametry uživatelského rozhraní na parametry algoritmu – např. u filtru, kdy je nutné jeho koeficienty (tj. parametry algoritmu) získat výpočtem z hodnoty mezního kmitočtu, zisku a šířky pásma (tj. z parametrů uživatelského rozhraní).
5) Funkce pro implementaci algoritmu číslicového zpracování signálů
   - `processReplacing`: pro zpracování zvukových dat ve formátu float (32bit)
   - `processDoubleRaplacing`: pro zpracování zvukových dat ve formátu double (64bit)

Aplikační rozhraní VST

Aplikační rozhraní VST je popsáno v originální HTML dokumentaci, kterou otevřete pomocí souboru „index.html“ v kořenovém adresáři šablony. Pro základní práci se šablonou není nutné tento dokument prostudovat, vystačíte s tímto textem. Hodit se vám ale může funkce `float getSampleRate()`, která vrací aktuální vzorkovací kmitočet.

Užitečné funkce šablony

Kromě funkcí standardního aplikačního rozhraní VST nabízí šablona VST plug-in modulu další užitečné funkce pro práci s parametry a podporu zpracování signálu.

- `float getRealValue(VstInt32 index)`: vrací aktuální reálnou hodnotu parametru.
- `unsigned long getProcessedSamples()`: vrací počet vzorků zpracovaných od spuštění zpracování do okamžiku posledního spuštění funkce `processReplacing`.
- `double getProcessingTime()`: vrací dobu zpracování od jeho spuštění do okamžiku posledního spuštění funkce `processReplacing`.
- `float *getInputBuffer(float **inputs,VstInt32 sampleFrames,VstInt32 channel)`: funkce vrací ukazatel na vstupní data daného kanálu, je nutné ji použít ve funkci `processReplacing`, pokud je definováno `def_tempInput` (viz šablona). Pro funkci `processDoubleReplacing` je definována přetížena pro datový typ `double`.
- `float*getOutputBuffer(float**outputs,VstInt32 sampleFrames,VstInt32 channel)`: funkce vrací ukazatel na výstupní data daného kanálu, je nutné ji použít ve funkci `processReplacing`, pokud je definováno `def_tempOutput` (viz šablona). Pro funkci `processDoubleReplacing` je definována přetížena pro datový typ `double`.
- `flushOutputBuffer(float **outputs,VstInt32 sampleFrames,VstInt32 channel)`: funkce provede zkopírování výstupních dat z dočasné vyrovnávací paměti do výstupní

paměti hostitelské aplikace, je nutné ji použít ve funkci `processReplacing`, pokud je definováno `def_tempOutput` (viz šablona). Pro funkci `processDoubleReplacing` je definována přetížena pro datový typ `double`.

Ladění plug-in modulu

Konfigurace *Debug* a *Debug with GTPlayer EDU* slouží k ladění programu. Plug-in modul není možné spouštět přímo, je nutné spustit hostitelskou aplikaci, ve které potom plug-in modul otevřete. V rámci projektu se neukládá nastavení ladicího programu (tzv. *Debuger*), proto je potřeba ho nastavit ručně pomocí dialogového okna nastavení projektu, které otevřete pomocí položky menu *Project* / *Properties* ... nebo klávesovou zkratkou Alt + F7. Zde v záložce *Debugging* zadejte do položky *Command* cestu k hostitelské aplikaci, kterou chcete při ladění plug-in modulu spouštět. Nastavení je individuální pro každou konfiguraci. Hostitelskou aplikaci je také možné zvolit při prvním pokusu o spuštění ve vývojovém prostředí (klávesová zkratka F5).

Ladění pomocí programu GT Player EDU

Pokud používáte pro ladění program GTPlayer EDU a konfiguraci *Debug with GTPlayer EDU*, zadejte v nastavení sestavovacího programu (viz výše) cestu „..\GTPlayerEDU\GT Player Express Edu.exe“.

Po spuštění programu se vámi vytvořený plug-in modul objeví jako první v seznamu plug-in modulů. Ten otevřete kliknutím na tlačítko *EDIT* a poté na libovolný z osmi slotů (s nápisem <*Empty*>, který znamená, že slot je zatím prázdný). Pokud jste neměnili nastavení sestavovacího programu, je váš plug-in modul v seznamu zobrazen jako „vst_temp“. Zpracování zvukového signálu je nutné aktivovat kliknutím na tlačítko *POWER*.

Pokud chcete jako vstupní signál použít zvukový soubor přehrávaný komponentou *Track Player*, je nutné v nastavení programu (tlačítko *OPTIONS*, položka *Preferences* ...) na záložce *Track Player* nastavit položku *Playback/record option* na *Before/clean*.

Ladění pomocí programu VST Plugin Analyser

Pro konfiguraci *Debug* můžete použít např. aplikaci *VST Plugin Analyser*, která je součástí balíčku šablony. Inicializační soubor aplikace je přednastaven tak, aby se automaticky otevřel plug-in modul vytvořený pomocí konfigurace *Debug*. V nastavení sestavovacího programu (viz výše) je nutné nastavit cestu „..\VST Plugin Analyser\VST Plugin Analyser.exe“ a pracovní adresář (položka *Working Directory*) „VST Plugin Analyser“.

Konečná kompilace plug-in modulu

Pokud máte plug-in modul odladěný, zvolte konfiguraci *Release* a proveďte znovu kompilaci. Sestavovací program (*Linker*) vytvoří knihovnu plug-in modulu v podadresáři `Release`, odkud ji můžete přesunout nebo zkopírovat do adresáře VST plug-in modulů hostitelské aplikace, ve které chcete váš plug-in modul používat (u aplikací firmy Steinberg adresář „C:\Program Files\Steinberg\VSTPlugins”, u většiny ostatních aplikací volitelný přímo v nastavení dané aplikace). Soubor knihovny můžete libovolně přejmenovat nebo požadované jméno přímo zadat do nastavení parametrů sestavovacího programu (menu *Project* / *Properties* ..., v záložce *Linker* položka *Output File*). Nastavení je individuální pro každou konfiguraci a jméno je potřeba zadat s celou cestou (lez využít maker, např. *$(OutDir)* pro výstupní adresář konfigurace).