### An Introduction to the CXL Reference Model

#### 一、运行环境

* GCC 12.2.0 及以上版本
* Make 3.81 及以上版本

**环境配置：**需将 gcc 和 make 添加至环境变量

**支持操作系统：**Linux、Windows

#### 调试、代码编写环境

**操作系统：**仅Windows

**开发工具：**Visual Studio Code（VS Code）

**VS Code 插件：**

**C/C++（必选）：**提供 C++ 语法检查及 GUI 调试支持（基于 GDB）。

**CodeGeeX: AI Coding Assistant（可选）：**AI代码补全。

**GitLens（可选）：**增强 Git 版本控制功能，支持本地及 GitHub 远程仓库管理。

#### 三、代码风格

本项目遵循以下代码风格规范，以保持一致性、可读性和可维护性：

##### 3.1文件规范

工程的文件组织如下：

│── .vscode/ # 配置 GUI 调试环境，内容由自动gpt生成

│── doc/ # 文档目录

│── include/ # 头文件目录（存放 .h头文件）

│── out/ # 编译输出目录（存放 .o目标文件）

│── pic/ # 图片资源目录

│── src/ # 源代码目录（存放 .cpp文件）

│── makefile # Makefile 编译脚本（自动化.cpp > .o > .exe的编译过程）

其中：对于 include/ 下的 .h 文件，和 src/ 下的 .cpp 文件：

**头文件 （.h文件）包含如下：**

* #ifndef，#define，#endif用以避免 .h 文件的重复包含。
* class 声明（class声明内包含有：成员变量的声明，非内敛成员函数的声明，内敛成员函数的声明和定义，enum class的声明）。

**源文件 （.cpp文件）包含如下：**

* class中非内敛成员函数的定义。

除了两个文件define.h 和main.cpp， .h 和 .cpp 文件命名保持一致，并全部小写。

例：s\_vsf\_line.h 和 s\_vsf\_line.cpp。

##### 3.2命名规范

###### 3.2.1类命名规范

单词间使用 \_ 分割，首字母大写，末尾加一个\_Type。

**例：**class Demo\_Test1\_Type{};

###### 3.2.2成员变量，成员函数命名规范

单词间使用 \_ 分割，全部小写。

**成员变量例：**int demo\_test2;

**成员函数例：**int demo\_test3(void);

###### 3.2.3 enum class命名规范

单词间使用 \_ 分割，首字母大写，末尾加一个\_Enum。

**例：**class enum S\_Demo\_Test4\_Enum{};

##### 3.3面向对象特性

在 C++ 工程的任何部分，应避免使用复杂的面向对象特性和语法，因为项目需求并不依赖这些特性。具体编码规范如下：

* 不使用全局变量，所有数据必须封装在 class 内，确保数据的可控性和封装性。
* 不使用继承和多态，所有 class 均为独立实现，避免不必要的层级关系。
* 不使用 friend 关键字。
* 不使用 protected 访问权限，仅使用 private、public 访问权限。
* 不定义新的数据结构，如 struct、typedef，仅使用 enum 来定义枚举类型。
* 不使用动态内存分配（new、delete），所有对象应为自动管理的栈对象或使用标准容器（如 std::vector、std::list）。

#### 3.4 class声明规范

###### 3.4.1类成员（包括成员变量和成员函数）类型限定

* 仅使用基本数据类型
* bool //具有(1bit)
* u32 //具有(32 bit)，即是unsigned int
* u64 //具有(64 bit)，即是unsigned long long
* u512
* 仅使用标准库字符串，即：
* std::string
* 仅使用标准库容器，即：
* std::list
* std::vector
* std::unordered\_map //项目工程内不用using namespace std
* 仅使用本项目内声明的其他 class

###### 3.4.2类成员函数的参数约定

**输入参数：**统一使用 const Class\_Type &，（避免拷贝）。

**输出参数：**统一使用 Class\_Type &，（避免返回值拷贝）。

**返回值类型：**为上述3.4.1所述的基本数据类型，或void。

##### 3.5 class构造和析构

* 要实现类的构造函数，用CLASS (T a) : a(0),这里的T只是基本数据类型，其他数据类型不用特意初始化。
* 不要实现类的析构函数。

方法（成员函数）的实现：

    std::list<S2M\_Snp\_type> s2m\_snp\_dpa;

    std::list<S2M\_NDR\_Type> s2m\_ndr\_dpa;

    std::list<S2M\_DRS\_Type> s2m\_drs\_dpa;

    main\_product(m2s\_rsp\_dpa, m2s\_rwd\_dpa, s2m\_snp\_dpa, s2m\_ndr\_dpa, s2m\_drs\_dpa, rd\_data, rd\_addr, wr);

之后，要写出s2m\_snp\_dpa, s2m\_ndr\_dpa, s2m\_drs\_dpa = main\_product(m2s\_rsp\_dpa, m2s\_rwd\_dpa)

Std::容器的实现

其他的方式不要定义push，也不要出现[]

容器成员的大小要被控制，当push（list）出现和 []（map）出现时，必须检查size，防止栈/堆内存溢出。