Framework API 参考

文档版本 01

发布日期 2020-05-09

华为技术有限公司

商标声明

HUAWEI和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

1 简介

本文介绍编译自定义算子时所依赖的Framework组件的接口（这部分接口是内部接口，用户不需要调用）、自定义算子插件开发涉及的接口，接口主要包括非类成员函数、类成员函数，同时本文还介绍了与接口相关的宏。

关于如何使用这些接口进行自定义算子插件的开发以及代码的编译运行指导，请参见《TE自定义算子开发指导》。

您可以在DDK包的安装目录下的“ddk/include/inc”目录下查看接口的定义文件。如果通过引导安装的方式同时安装Mind Studio和DDK，您可以使用Mind Studio安装用户登录Mind Studio服务器，在“~/tools/che/ddk/ddk/include/inc”路径下查看接口定义文件。每个接口对应的定义文件请参见具体的接口描述。

2 非类成员函数

2.1 custom_op_run 函数

函数功能

自定义算子的单算子调测入口函数。该函数在“custom\custom_op.h”中定义。

custom_op_run接口的返回值是ErrorInfo类型，包含错误码、错误描述。当错误码为0时，表示成功；当错误码为其它值时，表示失败。ErrorInfo类型的定义如下：

```
struct ErrorInfo
  {
    uint32_t    error_code;
    std::string error_msg;
  };
```

函数原型

ErrorInfo custom_op_run(const std::string& name, int32_t type, const std::string& bin_file,

const std::vector< std::string >& in_files,

const std::vector< std::string >& out_files,

const std::vector< uint32_t >& out_buf_sizes,

const std::vector< uint32_t >& workspace_sizes = std::vector<uint32_t>(),

const std::string& op_cfg_file = "", void *param = nullptr, int param_len = 0);

参数说明

参数	输入/输出	说明
name	输入	算子kernel名称。
type	输入	算子类型： ● 0 - TE_AICORE ● 1 -TE_AICPU ● 2 - AI CPU
bin_file	输入	算子二进制文件路径，包含文件名。 路径支持配置相对路径，此时二进制文件需存放在程序所在的目录下，如：**./*.bin**。
in_files	输入	输入数据文件路径，包含文件名。 路径支持配置相对路径，此时数据文件需存放在程序所在的目录下。
out_files	输入	输出数据文件路径，包含文件名。 路径支持配置相对路径，此时数据文件需存放在程序所在的目录下。
out_buf_sizes	输入	输出数据大小，单位是字节。
workspace_sizes	输入	workspace大小，单位是字节。
op_cfg_file	输入	用户指定的算子的输入输出描述信息。
param	输入	用户指定的算子参数结构体地址信息。
param_len	输入	用户指定的算子参数结构体的长度，使用sizeof(param)取结构体长度，单位是字节。

2.2 custom_op_compare 函数

函数功能

自定义算子的单算子校验接口，用于校验数据精度。该函数在“custom \custom_op.h”中定义。

custom_op_compare接口的返回值是ErrorInfo类型，包含错误码、错误描述。当错误码为0时，表示成功；当错误码为其它值时，表示失败。ErrorInfo类型的定义如下：

```
struct ErrorInfo
    {
        uint32_t    error_code;
        std::string error_msg;
    };
```

函数原型

ErrorInfo custom_op_compare(const std::string& expect_file, const std::string& actual_file,

int32_t data_type, float precision_deviation,

float statistical_discrepancy, bool& compare_result);

参数说明

参数	输入/输出	说明
expect_file	输入	期望数据文件，包含文件名
actual_file	输入	实际数据文件，包含文件名
data_type	输入	算子类型，目前支持FP32（0）、FP16（1）
precision_deviation	输入	单数据精度偏差，数值范围（0,1），偏差越小，精度越高。
statistical_discrepancy	输入	整个数据集的统计偏差，数值范围（0,1），偏差越小，精度越高。
compare_result	输出	比较结果， true：对比成功；false：对比失败

2.3 GetContext 函数

函数功能

获取OMG上下文，并返回。该函数在“framework\omg\omg_types.h”中定义。

GetContext接口的返回值是OmgContext类型，OmgContext类型的定义请参见“framework\omg\omg_types.h”。

函数原型

OmgContext& GetContext();

参数说明

无。

2.4 AutoMappingFn 函数

函数功能

自动映射回调函数。该函数在“framework\omg\register.h”中定义。

函数原型

Status AutoMappingFn(const google::protobuf::Message* op_src, ge::Operator& op);

参数说明

参数	输入/输出	说明
op_src	输入	原始转换前算子
op	输入	映射的算子

2.5 aicpu_run_func 函数

函数功能

AI CPU自定义算子的函数指针，函数名称可以自定义。该函数在“cce\customize.h”中定义。

函数原型

void (*aicpu_run_func)(opTensor_t **, void **, int32_t,

opTensor_t **, void **, int32_t, void *, rtStream_t);

参数说明

参数	输入/输出	说明

opTensor_t **	输入	输入描述符Tensor。
		typedef struct tagOpTensor
		{
		// real dim info
		opTensorFormat_t format;
		//char align1[4];
		opDataType_t data_type;
		//char align2[4];
		int32_t dim_cnt;
		int32_t mm; //lint !e148
		int32_t dim[CC_DEVICE_DIM_MAX]; //lint !e148
		} opTensor_t;

void **	输入	输入描述符Tensor的地址

参数	输入/输出	说明
int32_t	输入	输入描述符个数
opTensor_t **	输入	输出描述符Tensor
void **	输入	输出描述符Tensor的地址
int32_t	输入	输出描述符个数
void *	输入	属性handle地址
rtStream_t	输入	stream 标识

3 类成员函数

3.1 StatusFactory 类

该类函数在该函数在“custom\common\fmk_error_codes.h”中定义。

3.1.1 Instance 函数

函数功能

返回Status工厂类的实例对象。

函数原型

static StatusFactory* Instance();

参数说明

无

3.1.2 RegisterErrorNo 函数

函数功能

注册错误码。

函数原型

void RegisterErrorNo(uint32_t err, const std::string& desc);

参数说明

参数	输入/输出	说明
err	输入	错误码
desc	输入	错误码描述字符串

3.1.3 GetErrDesc 函数

函数功能

获取错误码描述。

函数原型

std::string GetErrDesc(uint32_t err);

参数说明

参数	输入/输出	说明
err	输入	错误码

3.1.4 StatusFactory 函数

函数功能

构造函数。

函数原型

StatusFactory();

参数说明

无

3.1.5 ~StatusFactory 函数

函数功能

析构函数。

函数原型

~StatusFactory();

参数说明

无

3.2 ErrorNoRegisterar 类

该类函数在该函数在“custom\common\fmk_error_codes.h”中定义。

3.2.1 ErrorNoRegisterar 函数

函数功能

构造函数。

函数原型

ErrorNoRegisterar(uint32_t err, const std::string& desc);

参数说明

参数	输入/输出	说明
err	输入	错误码
desc	输入	错误码描述字符串

3.2.2 ~ErrorNoRegisterar 函数

函数功能

析构函数。

函数原型

~ErrorNoRegisterar();

参数说明

无。

3.3 OpTypeContainer 类

该类函数在该函数在“custom\common\op_types.h”中定义。

3.3.1 Instance 函数

函数功能

获取OpTypeContainer工厂类的实例对象。

函数原型

static OpTypeContainer * Instance();

参数说明

无。

3.3.2 Register 函数

函数功能

将算子的类型保存到一个自定义的集合中。

函数原型

void Register(const std::string& op_type);

参数说明

参数	输入/输出	说明
op_type	输入	需要保存的算子类型

3.3.3 IsExisting 函数

函数功能

判断一个算子类型是否存在。

函数原型

bool IsExisting(const std::string& op_type);

参数说明

参数	输入/输出	说明
op_type	输入	需要查询的算子类型

3.3.4 OpTypeContainer 函数

函数功能

构造函数。

函数原型

OpTypeContainer();

参数说明

无。

3.4 OpTypeRegistrar 类

该类函数在该函数在“custom\common\op_types.h”中定义。

3.4.1 OpTypeRegistrar 函数

函数功能

构造函数，将一个算子的类型注册到Container中。

函数原型

OpTypeRegistrar(const std::string& op_type);

参数说明

参数	输入/输出	说明
op_type	输入	需要注册的算子类型

3.4.2 ~OpTypeRegistrar 函数

函数功能

析构函数。

函数原型

~OpTypeRegistrar();

参数说明

无

3.5 OpRegistrationData 类

该类函数在该函数在“framework\omg\register.h”中定义。

3.5.1 OpRegistrationData 函数

函数功能

构造函数。

函数原型

OpRegistrationData(const std::string& om_optype);

参数说明

参数	输入/输出	说明
om_optype	输入	算子类型

3.5.2 ~OpRegistrationData

函数功能

析构函数。

函数原型

~OpRegistrationData();

参数说明

无。

3.5.3 FrameworkType 函数

函数功能

设置框架类型。

函数原型

OpRegistrationData& FrameworkType(domi::FrameworkType fmk_type);

参数说明

参数	输入/输出	说明
fmk_type	输入	框架类型 0：caffe 3：tensorflow

3.5.4 OriginOpType 函数

函数功能

设置原始模型的算子类型或算子类型列表。

函数原型

- 设置算子类型列表：

  OpRegistrationData& OriginOpType (const std::initializer_list<std::string> & ori_optype_list);
- 设置算子类型：

  OpRegistrationData& OriginOpType (const std::string& ori_optype);

参数说明

参数	输入/输出	说明
ori_optype_list	输入	原始模型算子类型列表
ori_optype	输入	原始模型算子类型

3.5.5 ParseParamsFn 函数

函数功能

参数解析函数。

函数原型

OpRegistrationData& ParseParamsFn(ParseParamFunc parseParamFn);

参数说明

参数	输入/输出	说明
parseParamFn	输入	回调函数ParseParamFunc，请参见**回调函数ParseParamFunc**。

回调函数 ParseParamFunc

用户自定义并实现ParseParamFunc类函数，完成caffe模型参数和权值的转换，将结果填到Operator类中。

函数原型

Status ParseParamFunc(const Message* op_origin, ge::Operator& op_dest);

参数说明

参数	输入/输出	说明
op_origin	输入	protobuf格式的数据结构（来源于caffe模型的prototxt文件），包含算子参数信息。
op_dest	输出	适配昇腾AI处理器的离线模型的算子数据结构，保存算子信息。 关于Operator类，请参见《GE API参考》中的“Operator类接口”。

3.5.6 InferShapeAndTypeFn 函数

函数功能

维度推导函数。

函数原型

OpRegistrationData& InferShapeAndTypeFn(InferShapeFunc inferShapeFn);

参数说明

参数	输入/输出	说明
inferShapeFn	输入	回调函数InferShapeFunc，请参见**回调函数InferShapeFunc**。

回调函数 InferShapeFunc

用户自定义并实现InferShapeFunc类函数，用于获取算子的输出描述，包括输出shape信息、数据类型等张量描述信息。

函数原型

Status InferShapeFunc(const ge::Operator& op, vector<ge::TensorDesc>& v_output_desc);

参数说明

参数	输入/输出	说明
op	输入	适配昇腾AI处理器的离线模型的算子数据结构。 关于Operator类，请参见《GE API参考》中的“Operator类接口”。
v_output_desc	输出	存储算子的输出描述。 关于TensorDesc类，请参见《GE API参考》中的“TensorDesc类接口”。

3.5.7 UpdateOpDescFn 函数

函数功能

算子描述更新函数。

函数原型

OpRegistrationData& UpdateOpDescFn(UpdateOpDescFunc updateOpDescFn);

参数说明

参数	输入/输出	说明
updateOpDescFn	输入	回调函数UpdateOpDescFunc，参见**回调函数UpdateOpDescFunc**。

回调函数 UpdateOpDescFunc

用户自定义并实现UpdateOpDescFunc类函数，用于对算子信息数据进行更新。

函数原型

static Status UpdateOpDescFunc(ge::Operator& op);

参数说明

参数	输入/输出	说明
op	输入/输出	适配昇腾AI处理器的离线模型的算子数据结构，保存算子信息。 关于Operator类，请参见《GE API参考》中的“Operator类接口”。

3.5.8 GetWorkspaceSizeFn 函数

函数功能

获取work 空间大小。

函数原型

OpRegistrationData& GetWorkspaceSizeFn(GetWorkspaceSizeFunc getWorkspaceSizeFn);

参数说明

参数	输入/输出	说明
getWorkspaceSizeFn	输入	回调函数GetWorkspaceSizeFunc，参见**回调函数GetWorkspaceSizeFunc**。

回调函数 GetWorkspaceSizeFunc

用户自定义并实现GetWorkspaceSizeFunc类函数，完成算子对应的工作空间大小计算，并输出。

函数原型

Status GetWorkspaceSizeFn(domi::Status (const ge::Operator& op, std::vector<int64_t>& size));

参数说明

参数	输入/输出	说明
op	输入	需要计算工作空间的算子对象。 关于Operator类，请参见《GE API参考》中的“Operator类接口”。
size	输出	工作空间大小，该值需要用户计算并提供。

3.5.9 TEBinBuildFn 函数

函数功能

build回调函数。

函数原型

OpRegistrationData& TEBinBuildFn(BuildTeBinFunc buildTeBinFn);

参数说明

参数	输入/输出	说明
buildTeBinFn	输入	回调函数BuildTeBinFunc，参见**回调函数BuildTeBinFunc**。

回调函数 BuildTeBinFunc

用户自定义并实现BuildTeBinFunc类函数，用于构建算子二进制文件。

函数原型

virtual Status BuildTeBinFunc(const ge::Operator& op，TEBinInfo& teBinInfo);

参数说明

参数	输入/输出	说明
op	输入	适配昇腾AI处理器的离线模型的算子数据结构，保存算子信息。 关于Operator类，请参见《GE API参考》中的“Operator类接口”。

teBinInfo	输出	存储自定义算子二进制文件路径和ddk描述的数据。
		struct TEBinInfo
		{
		std::string bin_file_path; //自动从json文件binFileName字段获取。为了兼容以前用户编写的用例，字段不删除。
		std::string json_file_path;
		std::string ddk_version;
		};

3.5.10 ImplyType 函数

函数功能

设置算子执行方式。

函数原型

OpRegistrationData& ImplyType(domi::ImplyType imply_type);

参数说明

参数	输入/输出	说明

imply_type	输入	算子执行方式。
		enum class ImplyType : unsigned int
		{
		BUILDIN = 0,// 内置算子，由OME正常执行
		TVM, // 编译成tvm bin文件执行
		CUSTOM, // 由用户自定义计算逻辑，通过CPU执行
		AI_CPU, // AICPU 自定义算子类型
		INVALID = 0xFFFFFFFF,
		};

3.5.11 Formats 函数

函数功能

设置算子支持的输入/输出数据格式。

函数原型

```
OpRegistrationData& Formats(
const std::initializer_list<domi::tagDomiTensorFormat>& input_formats,
const std::initializer_list<domi::tagDomiTensorFormat>& output_formats);
OpRegistrationData& Formats(
const domi::tagDomiTensorFormat& input_format,
const domi::tagDomiTensorFormat& output_format);
```

参数说明

参数	输入/输出	说明
input_format	输入	输入支持的数据格式
output_format	输入	输出支持的数据格式

3.5.12 WeightFormats 函数

函数功能

设置算子权重支持的数据格式。

函数原型

```
OpRegistrationData& WeightFormats(
const std::initializer_list<domi::tagDomiTensorFormat>& weight_formats);
```

参数说明

参数	输入/输出	说明
weight_formats	输入	权重支持的数据格式列表： typedef enum tagDomiTensorFormat { DOMI_TENSOR_NCHW = 0, /**< NCHW */ DOMI_TENSOR_NHWC, /**< NHWC */ DOMI_TENSOR_ND, /**< Nd Tensor */ DOMI_TENSOR_NC1HWC0, /**< NC1HWC0 */ DOMI_TENSOR_FRACTAL_Z, /**< FRACTAL_Z */ DOMI_TENSOR_NC1C0HWPAD, DOMI_TENSOR_NHWC1C0, DOMI_TENSOR_FSR_NCHW, DOMI_TENSOR_FRACTAL_DECONV, DOMI_TENSOR_BN_WEIGHT, DOMI_TENSOR_CHWN, /*Android NN Depth CONV*/ DOMI_TENSOR_FILTER_HWCK, /* filter input tensor format */ DOMI_TENSOR_RESERVED } domiTensorFormat_t;

3.5.13 Finalize 函数

函数功能

创建所有支持的算子的parser和builder函数，成功则返回true，否则返回false。

函数原型

bool Finalize();

参数说明

无。

3.6 OpRegistry 类

该类函数在该函数在“framework\omg\register.h”中定义。

3.6.1 Instance 函数

函数功能

获取OpRegistry类对象的实例。

函数原型

static OpRegistry* Instance();

参数说明

无。

3.6.2 Register 函数

函数功能

注册一个OpRegistrationData，成功则返回true，否则返回false。

函数原型

bool Register(const OpRegistrationData& reg_data);

参数说明

参数	输入/输出	说明
reg_data	输入	算子注册信息

3.6.3 GetImplyType 函数

函数功能

获取算子执行类型。

函数原型

domi::ImplyType GetImplyType(const std::string& op_type);

参数说明

参数	输入/输出	说明
op_type	输入	需要查询的算子类型

3.6.4 GetOpTypeByImplyType 函数

函数功能

查询指定执行类型的所有算子的名称。

函数原型

```
void GetOpTypeByImplyType(std::vector<std::string>& vec_op_type,const
domi::ImplyType& imply_type);
```

参数说明

参数	输入/输出	说明
vec_op_type	输出	保存查询的算子名称信息
imply_type	输入	算子的执行类型

3.6.5 GetFormats 函数

函数功能

查询op_type这个算子支持的输入输出数据类型。

函数原型

```
void GetFormats(const std::string& op_type,
std::vector<domi::tagDomiTensorFormat>&
input_format_vector,std::vector<domi::tagDomiTensorFormat>&
output_format_vector);
```

参数说明

参数	输入/输出	说明
op_type	输入	算子类型名称
input_format_vector	输出	支持的输入数据类型列表
output_format_vector	输出	支持的输出数据类型列表

3.6.6 GetWeightFormats 函数

函数功能

查询op_type这个算子支持的权重数据类型。

函数原型

```
void GetWeightFormats(const std::string& op_type,
std::vector<domi::tagDomiTensorFormat>& format_vector);
```

参数说明

参数	输入/输出	说明
op_type	输入	算子类型名称
format_vector	输出	支持的权重数据类型列表

3.6.7 GetParseParamFunc 函数

函数功能

获取参数解析回调函数。

函数原型

```
domi::ParseParamFunc GetParseParamFunc(const std::string& op_type);
```

参数说明

参数	输入/输出	说明
op_type	输入	算子类型

3.6.8 GetInferShapeFunc 函数

函数功能

获取推导维度回调函数。

函数原型

```
domi::InferShapeFunc GetInferShapeFunc(const std::string& op_type);
```

参数说明

参数	输入/输出	说明
op_type	输入	算子类型

3.6.9 GetGetWorkspaceSizeFunc 函数

函数功能

获取workspace大小回调函数。

函数原型

```
domi::GetWorkspaceSizeFunc GetGetWorkspaceSizeFunc(const std::string&
op_type);
```

参数说明

参数	输入/输出	说明
op_type	输入	算子类型

3.6.10 GetUpdateOpDescFunc 函数

函数功能

获取更新算子描述回调函数。

函数原型

```
domi::UpdateOpDescFunc GetUpdateOpDescFunc(const std::string& op_type);
```

参数说明

参数	输入/输出	说明
op_type	输入	算子类型

3.6.11 GetBuildTeBinFunc 函数

函数功能

获取build回调函数。

函数原型

domi::BuildTeBinFunc GetBuildTeBinFunc(const std::string& op_type);

参数说明

参数	输入/输出	说明
op_type	输入	算子类型

3.6.12 GetTransWeightFunc 函数

函数功能

获取转换权值回调函数。

函数原型

domi::TransWeightFunc GetTransWeightFunc(const std::string& op_type);

参数说明

参数	输入/输出	说明
op_type	输入	算子类型

3.7 OpReceiver 类

该类函数在该函数在“framework\omg\register.h”中定义。

3.7.1 OpReceiver 函数

函数功能

构造函数。

函数原型

OpReceiver(OpRegistrationData& reg_data);

参数说明

参数	输入/输出	说明
reg_data	输入	需要注册的算子信息

3.7.2 ~OpReceiver 函数

函数功能

析构函数。

函数原型

~OpReceiver();

参数说明

无。

4 宏定义

4.1 错误码生成宏

宏功能

调用DEF_ERRORNO生成指定模块的错误码。

宏原型

DEF_ERRORNO(sysid, modid, name, value, desc)

参数说明

参数	输入/输出	说明
sysid	输入	系统id
modid	输入	模块id
name	输入	错误码名称

参数	输入/输出	说明
value	输入	错误码的实际值
desc	输入	错误码描述字符串

4.2 COMMON 模块错误码生成宏

宏功能

调用DEF_ERRORNO_COMMON生成common模块的错误码。

宏原型

DEF_ERRORNO_COMMON(name, value, desc)

参数说明

参数	输入/输出	说明
name	输入	错误码名称
value	输入	错误码的实际值
desc	输入	错误码描述字符串

4.3 OMG 模块错误码生成宏

宏功能

调用DEF_ERRORNO_OMG生成OMG模块的错误码。

宏原型

DEF_ERRORNO_OMG(name, value,desc)

参数说明

参数	输入/输出	说明
name	输入	错误码名称
value	输入	错误码的实际值
desc	输入	错误码描述字符串

4.4 OME 模块错误码生成宏

宏功能

调用DEF_ERRORNO_OME生成OME模块的错误码。

宏原型

DEF_ERRORNO_OME(name, value,desc)

参数说明

参数	输入/输出	说明
name	输入	错误码名称
value	输入	错误码的实际值
desc	输入	错误码描述字符串

4.5 CALIBRATION 模块错误码生成宏

宏功能

调用DEF_ERRORNO_CALIBRATION生成量化模块的错误码。

宏原型

DEF_ERRORNO_CALIBRATION(name, value, desc)

参数说明

参数	输入/输出	说明
name	输入	错误码名称
value	输入	错误码的实际值
desc	输入	错误码描述字符串

4.6 获取错误码描述宏

宏功能

调用GET_ERRORNO_STR获取错误码描述。

宏原型

GET_ERRORNO_STR(value)

参数说明

参数	输入/输出	说明
value	输入	错误码的值

4.7 算子类型注册宏

宏功能

对一种算子类型进行注册，主要是声明一个常量字符串,该字符串就是算子类型的名称，然后将其保存到一个set中。

宏原型

REGISTER_OPTYPE_DECLARE(var_name, str_name)

REGISTER_OPTYPE_DEFINE(var_name, str_name)

参数说明

参数	输入/输出	说明
var_name	输入	常量字符串名称
str_name	输入	算子类型名称

4.8 算子类型是否存在宏

宏功能

检查一个算子类型是否存在。

宏原型

IS_OPTYPE_EXISTING(str_name)

参数说明

参数	输入/输出	说明
str_name	输入	算子类型名称

4.9 算子注册宏

4.9.1 REGISTER_CUSTOM_OP 宏

宏功能

按指定名称注册算子。

宏原型

REGISTER_CUSTOM_OP(name)

参数说明

参数	输入/输出	说明
name	输入	算子名称

4.10 对外接口标记宏

4.10.1 FMK_FUNC_HOST_VISIBILITY

宏功能

标记Host侧对外暴露接口，使用该宏标记接口在动态链接库中可见并可被调用。

宏原型

FMK_FUNC_HOST_VISIBILITY

参数说明

无

4.10.2 FMK_FUNC_DEV_VISIBILITY

宏功能

标记Device侧对外暴露接口，使用该宏标记接口在动态链接库中可见并可被调用。

宏原型

FMK_FUNC_DEV_VISIBILITY

参数说明

无

5 附录

5.1 修订记录

文档版本	发布日期	修改说明
01	2020-05-09	第一次正式发布。