Ascend 310
V100R001

# AI CPU API 参考

| | |
|---|---|
| 文档版本 | 01 |
| 发布日期 | 2019-03-12 |

华为技术有限公司

**商标声明**


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。

# 华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

# 前 言

## 概述

本文档用于描述Ascend 310项目AI CPU算子的接口。

## 读者对象

本文档主要描述AI CPU对外提供的接口，适合Ascend 310 V100R001项目相关开发人员、测试人员。

## 符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

| 符号 | 说明 |
| --- | --- |
|  危险 | 用于警示紧急的危险情形，若不避免，将会导致人员死亡或严重的人身伤害。 |
|  警告 | 用于警示潜在的危险情形，若不避免，可能会导致人员死亡或严重的人身伤害。 |
|  小心 | 用于警示潜在的危险情形，若不避免，可能会导致中度或轻微的人身伤害。 |
|  注意 | 用于传递设备或环境安全警示信息，若不避免，可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。<br>“注意”不涉及人身伤害。 |
|  说明 | 用于突出重要/关键信息、最佳实践和小窍门等。<br>“说明”不是安全警示信息，不涉及人身、设备及环境伤害。 |

# 目 录



  

  

# 1 API 总览

## 关于本章



## 1.1 API 定义

AI CPU算子Device接口如**表1-1**所示。

**表 1-1** AI CPU 算子 Device 接口

| 接口 | 说明 |
| --- | --- |
| ccblas_device_hasum | 该函数提供对向量的绝对值求和。 |
| ccblas_device_ihamin | 该函数提供求向量元素绝对值最小的索引。 |
| ccblas_device_ihamax | 该函数提供求向量中最大值的索引。 |
| ccblas_device_haxpy | 该函数提供常量和向量x相乘，加向量y的过程。 |
| ccblas_device_hcopy | 该函数提供向量拷贝的过程。 |
| ccblas_device_hnrm2 | 该函数提供欧几里得范数的求解过程。 |
| ccblas_device_hscal | 该函数提供向量与标量的乘法。 |
| ccblas_device_hrotg | 该函数提供Givens旋转矩阵求解。 |
| ccblas_device_hrotmg | 该函数提供modified givens rotation matix H的构造。 |
| ccblas_device_hswap | 该函数提供两个向量交换的过程。 |
| ccblas_device_hgemv | 该函数提供矩阵与向量乘法。 |

| 接口 | 说明 |
| --- | --- |
| ccblas_device_hgemv_ex | 该函数实现矩阵与向量乘法。 |
| ccblas_device_hger | 该函数提供实现矩阵的rank-1更新。 |
| ccblas_device_htrsv | 该函数提供三角线性方程的求解。 |
| ccblas_device_htbsv | 该函数提供三角带型矩阵的求解。 |
| ccblas_device_htpsv | 该函数提供op(A)x=b方程求x的过程。 |
| ccblas_device_hgemm | 该函数提供矩阵与矩阵乘法。 |
| ccblas_device_hgemm_ex | 该函数实现矩阵与矩阵乘法。 |
| ccblas_device_hdot | 该函数实现了两个向量的内积。 |

**说明**

- 所有矩阵数据都是按列存储。
- 所有输入向量长度需要满足运算长度。

# 1.2 枚举定义

## 1.2.1 ccblasFillMode_t

| 定义 | 成员说明 |
| --- | --- |
| typedef enum {<br>CCBLAS_FILL_MODE_LOWER = 0,<br>CCBLAS_FILL_MODE_UPPER = 1<br>} ccblasFillMode_t; | ● CCBLAS_FILL_MODE_LOWER：下三角矩阵。<br>● CCBLAS_FILL_MODE_UPPER：上三角矩阵。 |

## 1.2.2 ccblasOperation_t

| 定义 | 成员说明 |
| --- | --- |
| typedef enum {<br>CCBLAS_OP_N = 0,<br>CCBLAS_OP_T = 1,<br>} ccblasOperation_t; | ● CCBLAS_OP_N：非转置。<br>● CCBLAS_OP_T：转置。 |

  

### 1.2.3 ccblasDiagType_t

| 定义 | 成员说明 |
| --- | --- |
| typedef enum {<br>CCBLAS_DIAG_NON_UNIT = 0,<br>CCBLAS_DIAG_UNIT = 1<br>} ccblasDiagType_t; | ● CCBLAS_DIAG_NON_UNIT：对角线元素不全为1。<br>● CCBLAS_DIAG_UNIT：对角线元素全为1。 |

## 1.3 错误码定义

  

## 1.3.1 DEVICE 层错误码定义

| 定义 | 成员说明 |
| --- | --- |
| typedef enum CCBLAS_DEV_ERR_CODE<br>{<br>CCBLAS_DEV_STATUS_SUCCESS = 0,<br>CCBLAS_DEV_STATUS_M_ERR,<br>CCBLAS_DEV_STATUS_N_ERR,<br>CCBLAS_DEV_STATUS_K_ERR,<br>CCBLAS_DEV_STATUS_INCX_ERR,<br>CCBLAS_DEV_STATUS_INCY_ERR = 5,<br>CCBLAS_DEV_STATUS_ALPHA_ERR,<br>CCBLAS_DEV_STATUS_BETA_ERR,<br>CCBLAS_DEV_STATUS_LDA_ERR,<br>CCBLAS_DEV_STATUS_LDB_ERR,<br>CCBLAS_DEV_STATUS_LDC_ERR = 10,<br>CCBLAS_DEV_STATUS_UPLO_ERR,<br>CCBLAS_DEV_STATUS_DIAG_ERR,<br>CCBLAS_DEV_STATUS_TRANS_ERR,<br>CCBLAS_DEV_STATUS_TRANSA_ERR,<br>CCBLAS_DEV_STATUS_TRANSB_ERR = 15,<br>CCBLAS_DEV_STATUS_X_NULL_ERR,<br>CCBLAS_DEV_STATUS_Y_NULL_ERR,<br>CCBLAS_DEV_STATUS_A_NULL_ERR, | CCBLAS_DEV_STATUS_SUCCESS：函数执行成功。<br>CCBLAS_DEV_STATUS_M_ERR：入参M错误。<br>CCBLAS_DEV_STATUS_N_ERR：入参N错误。<br>CCBLAS_DEV_STATUS_K_ERR：入参K错误。<br>CCBLAS_DEV_STATUS_INCX_ERR：入参INCX错误。<br>CCBLAS_DEV_STATUS_INCY_ERR：入参INCY错误。<br>CCBLAS_DEV_STATUS_ALPHA_ERR：入参ALPHA错误。<br>CCBLAS_DEV_STATUS_BETA_ERR：入参BETA错误。<br>CCBLAS_DEV_STATUS_LDA_ERR：入参LDA错误。<br>CCBLAS_DEV_STATUS_LDB_ERR：入参LDB错误。<br>CCBLAS_DEV_STATUS_LDC_ERR：入参LDC错误。<br>CCBLAS_DEV_STATUS_UPLO_ERR：入参UPLO错误。<br>CCBLAS_DEV_STATUS_DIAG_ERR：入参DIAG错误。<br>CCBLAS_DEV_STATUS_TRANS_ERR：入参TRANS错误。<br>CCBLAS_DEV_STATUS_TRANSA_ERR：入参TRANSA错误。<br>CCBLAS_DEV_STATUS_TRANSB_ERR：入参TRANSB错误。<br>CCBLAS_DEV_STATUS_X_NULL_ERR：入参X为NULL非法。<br>CCBLAS_DEV_STATUS_Y_NULL_ERR：入参Y为NULL非法。<br>CCBLAS_DEV_STATUS_A_NULL_ERR：入参A为NULL非法。<br>CCBLAS_DEV_STATUS_B_NULL_ERR：入参B为NULL非法。<br>CCBLAS_DEV_STATUS_C_NULL_ERR：入参C为NULL非法。<br>CCBLAS_DEV_STATUS_a_NULL_ERR：入参a为NULL非法。 |

  

| 定义 | 成员说明 |
| --- | --- |
| CCBLAS_DEV_STATUS_B_NULL_ERR,<br>CCBLAS_DEV_STATUS_C_NULL_ERR = 20,<br>CCBLAS_DEV_STATUS_a_NULL_ERR,<br>CCBLAS_DEV_STATUS_b_NULL_ERR,<br>CCBLAS_DEV_STATUS_c_NULL_ERR,<br>CCBLAS_DEV_STATUS_s_NULL_ERR,<br>CCBLAS_DEV_STATUS_d1_NULL_ERR = 25,<br>CCBLAS_DEV_STATUS_d2_NULL_ERR,<br>CCBLAS_DEV_STATUS_x1_NULL_ERR,<br>CCBLAS_DEV_STATUS_y1_NULL_ERR,<br>CCBLAS_DEV_STATUS_param_NULL_ERR,<br>CCBLAS_DEV_STATUS_RESULT_NULL_ERR = 30,<br>CCBLAS_DEV_STATUS_DONON_ERR,<br>CCBLAS_DEV_STATUS_ERR_MAX,<br>}ENUM_CCBLAS_DEV_ERR_CODE; | CCBLAS_DEV_STATUS_b_NULL_ERR：入参b为NULL非法。<br>CCBLAS_DEV_STATUS_c_NULL_ERR：入参c为NULL非法。<br>CCBLAS_DEV_STATUS_s_NULL_ERR：入参s为NULL非法。<br>CCBLAS_DEV_STATUS_d1_NULL_ERR：入参d1为NULL非法。<br>CCBLAS_DEV_STATUS_d2_NULL_ERR：入参d2为NULL非法。<br>CCBLAS_DEV_STATUS_x1_NULL_ERR：入参x1为NULL非法。<br>CCBLAS_DEV_STATUS_y1_NULL_ERR：入参y1为NULL非法。<br>CCBLAS_DEV_STATUS_param_NULL_ERR：入参param为NULL非法。<br>CCBLAS_DEV_STATUS_RESULT_NULL_ERR：入参result为NULL非法。 |

**说明**

#define AICPU_EID(mid, eid) ((0xFFFF0000 & (mid << 16)) | (0xFFFF & (eid)))

每个算子返回的实际错误码为本身的module id和错误码两部分按上述规则组成。

  

# 2 AI CPU 算子 Device 接口

## 关于本章



## 2.1 Level1 接口

### 2.1.1 ccblas_device_hasum

#### 函数功能

该接口实现对向量的绝对值求和。

#### 函数格式

**int32_t ccblas_device_hasum(**

**int32_t n,**

**const half_float* x,**

**int32_t incx,**

**half_float* result)**

#### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量元素个数 | >0的整数 |

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **const half_float*x** | 输入数据起始地址 | 不能为NULL<br>**说明**<br>x的长度>=1+(n-1)*abs(incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | >0的整数 |
| **half_float * result** | 输出绝对值求和结果 | 不能为NULL<br>**说明**<br>当incx<=0或n<=0时，返回的result值为0.0 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**。

函数执行失败时返回错误码。

### 相关函数

无。

## 2.1.2 ccblas_device_ihamin

### 函数功能

该接口实现求向量元素绝对值最小的索引。

### 函数格式

**int32_t ccblas_device_ihamin(**

**int32_t n,**

**const half_float *x,**

**int32_t incx,**

**int32_t* result)**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量元素的总的个数 | >0的整数 |
| **const half_float*x** | 输入向量起始地址 | 不能为NULL<br>**说明**<br>x的长度>=(1+（n-1）*incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 取值>=1 |

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t * result** | 输出向量绝对值中最小值的索引 | 不能为NULL<br>**说明**<br>当incx<=0或n<=0时，返回的result值为0 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.1.3 ccblas_device_ihamax

### 函数功能

该接口实现求向量中最大值的索引。

### 函数格式

**int32_t ccblas_device_ihamax(**

**int32_t n,**

**const half_float *x,**

**int32_t inc_x,**

**int32_t *result)**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量元素个数 | 取值范围>0 |
| **const half_float*x** | 输入数据起始地址 | 不能为NULL<br>**说明**<br>x的长度>=(1+（n-1）*incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 取值>=1 |
| **int32_t * result** | 输出向量中绝对值的最大值索引 | 不能为NULL<br>**说明**<br>当incx<=0或n<=0时，返回的result值为0 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.1.4 ccblas_device_haxpy

### 函数功能

该函数实现了常量和向量x相乘，加向量y的过程。

### 函数格式

**int32_t ccblas_device_haxpy(**

**int32_t n,**

**const half_float alpha,**

**const half_float * x,**

**int32_t incx,**

**half_float * y,**

**int32_t incy))**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量元素个数 | >0的整数 |
| **half_float alpha** | 向量x的标量值 | 任意值 |
| **const half_float * x:** | 输入向量x首地址 | 不能为NULL<br>**说明**<br>x的长度>=1+(n-1)*abs(incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 任意值，但需要满足和x之间的关系 |
| **half_float * y** | 输入向量y首地址<br>同时也是输出向量y的首地址 | 不能为NULL<br>**说明**<br>y的长度>=1+(n-1)*abs(incy) |
| **int32_t incy** | 向量y连续两个元素之间的步长 | 任意值，但需要满足和y之间的关系 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.1.5 ccblas_device_hcopy

### 函数功能

该函数实现向量拷贝的过程。

### 函数格式

**ccblas_device_hcopy(**

**int32_t n,**

**const half_float *x,**

**int32_t incx,**

**half_float *y,**

**int32_t incy)**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量元素个数 | 取值范围>0 |
| **const half_float * x** | 输入数据起始地址 | 不能为NULL<br>**说明**<br>x的长度>=(1+n*abs(incx)) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 任意整数 |
| **half_float* y** | 输出数据起始地址 | 不能为NULL<br>**说明**<br>y的长度>=(1+n*abs(incy)) |
| **int32_t incy** | 向量y连续两个元素之间的步长 | 任意整数 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

  

## 相关函数

无。

# 2.1.6 ccblas_device_hnrm2

## 函数功能

该函数实现了欧几里得范数的求解过程。

## 函数格式

**int32_t ccblas_device_hnrm2(**

**int32_t n,**

**const half_float *x,**

**int32_t inc_x,**

**half_float* result)**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量元素个数 | >0的整数 |
| **const half_float * x** | 输入数据起始地址 | 不能为NULL<br>**说明**<br>x的长度>=（1+(n-1)*abs(incx)） |
| **int32_t incx** | 向量x连续两个元素之间的步长。 | >=1的整数 |
| **half_float * result** | 计算结果 | 不能为NULL<br>**说明**<br>当n<=0或incx<=0时，*result=0.0 |

## 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

## 相关函数

无。

## 2.1.7 ccblas_device_hscal

### 函数功能

该函数实现向量与标量的乘法。

### 函数格式

**int32_t ccblas_device_hscal(**

**int32_t n,**

**half_float alpha,**

**half_float* x,**

**int32_t incx)**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量元素个数 | 取值范围>0 |
| **half_float * x** | 输入数据x起始地址<br>同时也是输出数据的起始地址<br>**说明**<br>计算结果为对应位置上的数与scale的乘积，没有被操作的数保持原值不变 | 不能为NULL<br>**说明**<br>x长度>=(1+（n-1）*incx) |
| **half_float alpha** | 乘法公式的中标量，scale系数 | 任意值 |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 取值范围>=1 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.1.8 ccblas_device_hrotg

### 函数功能

该函数实现Givens旋转矩阵求解。

### 函数格式

**int32_t ccblas_device_hrotg(**

**half_float* a,**

**half_float* b,**

**half_float* c,**

**half_float* s);**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **half_float * a** | 输入x-y平面上二维向量的横坐标<br>同时也是输出 | 不能为NULL |
| **half_float * b** | 输入x-y平面上二维向量的纵坐标<br>同时也是输出 | 不能为NULL |
| **half_float * c** | 输出旋转矩阵的cosin值 | 不能为NULL |
| **half_float * s** | 输出旋转矩阵的sin值 | 不能为NULL |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.1.9 ccblas_device_hrotmg

### 函数功能

该函数实现了modified givens rotation matix H的构造。

### 函数格式

**int32_t ccblas_device_hrotmg(**

**half_float *d1,**

**half_float *d2,**

**half_float *x1,**

**const half_float *y1,**

**half_float *param)**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **half_float* d1** | 用于辅助计算H矩阵，在函数中会被重写。 | 不能为NULL。 |
| **half_float* d2** | 用于辅助计算H矩阵，在函数中会被重写。 | 不能为NULL。 |
| **half_float* x1** | 用于计算H矩阵，在函数中会被重写。 | 不能为NULL。 |
| **const half_float* y1** | 用于计算H矩阵，在函数中不会被重写。 | 不能为NULL。 |
| **half_float* param** | 在函数中会被重写，保存flag和H矩阵。<br>其中param[0]保存flag，param[1]保存h11，param[2]保存h21，param[3]保存h12，param[4]保存h22。<br>● 对于param[0]=-2，param[1] – param[4]不会在程序中重写，默认的对应值为1 0 0 1。<br>● 对于param[0]=-1，param[1] - param[4]会在程序中重写。<br>● 对于param[0]=0，param[2]、param[3]会在程序中重写，param[1]、param[4]默认为1 1。<br>● 对于param[0]=1，param[1]、param[4]会在程序中重写，param[2]、param[3]默认为1 -1。<br>**注意**<br>这里的默认不是指param会赋予一个默认值，而是当函数返回后，用户看到param[0]的值后，就应该知道默认位置处的值，在param里对应位置处的值在调用前后是没有变化的。 | 不能为NULL。 |

## 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

## 相关函数

无。

# 2.1.10 ccblas_device_hswap

## 函数功能

该函数实现了两个向量交换的过程。

### 函数格式

**int32_t ccblas_device_hswap(**

**int32_t n,**

**half_float *x,**

**int32_t incx,**

**half_float *y,**

**int32_t incy);**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量运算的个数 | >0的整数 |
| **half_float * x** | 向量x首地址。既是输入指针又是输出指针 | 不能为NULL<br>**说明**<br>x的长度=1+(n-1)*abs(incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 任意值 |
| **half_float * y** | 向量y首地址。既是输入指针又是输出指针 | 不能为NULL<br>**说明**<br>y的长度=1+(n-1)*abs(incy) |
| **int32_t incy** | 向量y连续两个元素之间的步长 | 任意值 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.1.11 ccblas_device_hdot

### 函数功能

该函数实现了两个向量的内积。

### 函数格式

**int32_t ccblas_device_hdot(**

**int32_t n,**

**const half_float *x,**

**int32_t incx,**

**const half_float *y,**

**int32_t incy,**

***half_float *result);***

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t n** | 向量运算的个数 | >0的整数 |
| **half_float * x** | 向量x首地址 | 不能为NULL<br>**说明**<br>x的长度>=1+(n-1)*abs(incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 任意值 |
| **half_float * y** | 向量y首地址 | 不能为NULL<br>**说明**<br>y的长度>=1+(n-1)*abs(incy) |
| **int32_t incy** | 向量y连续两个元素之间的步长 | 任意值 |
| **half_float * result** | 输出结果的地址 | 不能为NULL<br>**说明**<br>当n<=0时，*result=0.0 |

### 返回值

函数执行成功时返回**CCBLAS_DEV_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

# 2.2 Level2 接口

## 2.2.1 ccblas_device_hgemv

### 函数功能

该函数实现矩阵与向量乘法。

  

## 函数格式

**int32_t ccblas_device_hgemv(**

**ccblasOperation_t trans,**

**int32_t m,**

**int32_t n,**

**const half_float alpha,**

**const half_float* A,**

**int32_t lda,**

**const half_float* x,**

**int32_t incx,**

**const half_float beta,**

**half_float* y,**

**int32_t incy);**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **ccblasOperation_t trans** | 输入为：CCBLAS_OP_N或CCBLAS_OP_T，其中CCBLAS_OP_T表示转置矩阵 | 请参见**1.2.2 ccblasOperation_t** |
| **int32_t m** | 矩阵A的行数 | 大于0的整数 |
| **int32_t n** | 矩阵A的列数 | 大于0的整数 |
| **const half_float alpha** | 标量，比例因子 | 任意值 |
| **const half_float* A** | 输入矩阵A的首地址<br>**说明**<br>维度为lda×n的数组，其中lda>=max(1,m)，以列形式存储 | 不能为NULL<br>**说明**<br>维度为lda×n的矩阵，其中lda>=max(1,m) |
| **int32_t lda** | 用于存储矩阵A的二维数组的第一个维度值 | lda>= max(1,m) |
| **const half_float* X** | 输入向量X的首地址 | 不能为NULL<br>**说明**<br>● 当trans==CCBLAS_OP_N时，至少有(1+(n-1)*abs(incx))个元素<br>● 其他情况，至少有(1+(m-1)*abs(incx))个元素 |

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t Incx** | 向量x连续两个元素之间的步长 | 非0整数 |
| **const half_float beta** | 标量，比例因子 | 任意值 |
| **half_float* y** | 输入和输出向量y的首地址<br>计算得到的结果会重写y | 不能为NULL<br>**说明**<br>● 当trans==CCBLAS_OP_N时，至少有(1+(m-1)*abs(incy))个元素<br>● 其他情况，至少有(1+(n-1)*abs(incy))个元素 |
| **int32_t incy** | 向量y连续两个元素之间的步长 | 非0整数 |

## 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回错误码。

## 相关函数

无。

# 2.2.2 ccblas_device_hgemv_ex

## 函数功能

该函数实现矩阵与向量乘法。

该接口专供TE使用。

## 函数格式

**int32_t ccblas_device_hgemv_ex(**

**int trans,**

**int32_t m,**

**int32_t n,**

**const __fp16 alpha,**

**const __fp16* A,**

**int32_t lda,**

**const __fp16* x,**

**int32_t incx,**

**const __fp16 beta,**

**__fp16* y,**

**int32_t incy)**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int trans** | 输入为：<br>● 0：表示CCBLAS_OP_N<br>● 1：CCBLAS_OP_T，其中CCBLAS_OP_T表示转置矩阵 | 请参见**1.2.2 ccblasOperation_t** |
| **int32_t m** | 矩阵A的行数 | 大于0的整数 |
| **int32_t n** | 矩阵A的列数 | 大于0的整数 |
| **const __fp16 alpha** | 标量，比例因子 | 任意值 |
| **const __fp16* A** | 输入矩阵A的首地址<br>**说明**<br>维度为lda × n的数组。其中lda>= max(1,m)，以列形式存储 | 不能为NULL<br>**说明**<br>维度为lda×n的矩阵。其中lda>=max(1,m) |
| **int32_t lda** | 用于存储矩阵A的二维数组的第一个维度值 | lda>=max(1,m) |
| **const __fp16* x** | 输入向量X的首地址 | 不能为NULL<br>**说明**<br>1.当trans==CCBLAS_OP_N时，至少有(1+(n-1)*abs(incx))个元素<br>2.其他情况，至少有(1+(m-1)*abs(incx))个元素 |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 非0整数 |
| **const __fp16 beta** | 标量，比例因子 | 任意值 |
| **__fp16* y** | 输入和输出向量y的首地址<br>计算得到的结果会重写y | 不能为NULL<br>**说明**<br>1.当trans==CCBLAS_OP_N时，至少有(1+(m-1)*abs(incy))个元素<br>2.其他情况，至少有(1+(n-1)*abs(incy))个元素 |
| **int32_t incy** | 向量y连续两个元素之间的步长 | 非0整数 |

### 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.2.3 ccblas_device_hger

### 函数功能

该函数实现矩阵的rank-1更新。

### 函数格式

**int32_t ccblas_device_hger(**

**int32_t m,**

**int32_t n,**

**const half_float alpha,**

**const half_float *x,**

**int32_t inc_x,**

**const half_float *y,**

**int32_t inc_y,**

**half_float *a,**

**int32_t lda)**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int32_t m** | 矩阵A的行数 | >0的整数 |
| **int32_t n** | 矩阵A的列数 | >0的整数 |
| **half_float alpha** | 标量，比例因子 | 任意值 |
| **const half_float * x** | 输入向量x数据起始地址 | 不能为NULL<br>**说明**<br>至少有(1+(m-1)*abs(incx))个元素 |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 非0值 |

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **const half_float * y** | 输入向量y起始地址 | 不能为NULL。<br>**说明**<br>至少有(1+(n-1)*abs(inc_y))个元素 |
| **int32_t incy** | 向量y连续两个元素之间的步长 | 非0值，但需要满足和y之间的关系 |
| **half_float * A** | 输入矩阵的起始地址<br>同时也是输出矩阵的起始地址 | 不能为NULL<br>**说明**<br>维度为lda×n的矩阵。其中lda>=max(1,m) |
| **int32_t lda** | 用于存储矩阵A的二维数组的第一个维度值 | lda>=max(1,m) |

## 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回错误码。

## 相关函数

无。

# 2.2.4 ccblas_device_htrsv

## 函数功能

该函数实现了三角线性方程的求解。

## 函数格式

**int32_t ccblas_device_htrsv(**

**ccblasFillMode_t uplo,**

**ccblasOperation_t trans,**

**ccblasDiagType_t diag,**

**int32_t n,**

**const half_float *A,**

**int32_t lda,**

**half_float *x,**

**int32_t incx)**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **ccblasFillMode_t uplo** | 指示三角矩阵A存储上半部分或下半部分，其他元素未引用或可从存储的部分进行推导 | ● CCBLAS_FILL_MODE_LOWER存储下半部分<br>● CCBLAS_FILL_MODE_UPPER存储上半部分 |
| **ccblasOperation_t trans** | 针对矩阵A的枚举类型定义 | ● CCBLAS_OP_N：表示不做任何操作<br>● CCBLAS_OP_T：表示对指定对象做转置操作 |
| **ccblasDiagType_t diag** | 指示主对角线上的元素是否是全1 | ● CCBLAS_DIAG_NON_UNIT：表示对角线有非1元素<br>● CCBLAS_DIAG_UNIT：表示对角线元素全为1 |
| **int32_t n** | 矩阵A的行数和列数 | 大于0的整数值，满足参数A的要求 |
| **const half_float * A** | 输入矩阵A的起始地址 | 不能为NULL<br>**说明**<br>维度为lda×n的矩阵，lda>=max(1,n) |
| **int32_t lda** | 用于存储矩阵A的二维数组的第一个维度值 | lda>= max(1,n) |
| **half_float * x** | 输入向量x的起始地址<br>同时是输出向量的起始地址 | 不能为NULL<br>**说明**<br>x的长度>=1+(n-1)*abs(incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长。 | 非0整数 |

## 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回错误码。

## 相关函数

无。

# 2.2.5 ccblas_device_htbsv

## 函数功能

该函数实现三角带型矩阵的求解。

## 函数格式

**int32_t ccblas_device_htbsv(**

**ccblasFillMode_t uplo,**

**ccblasOperation_t trans,**

**ccblasDiagType_t diag,**

**int32_t n,**

**int32_t k,**

**const half_float* a,**

**int32_t lda,**

**half_float* x,**

**int32_t incx)**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **ccblasFillMode_t uplo** | 指示三角矩阵A存储上半部分或下半部分，其他元素未引用或可从存储的部分进行推导 | ● CCBLAS_FILL_MODE_LOWER存储下半部分<br>● CCBLAS_FILL_MODE_UPPER存储上半部分 |
| **ccblasOperation_t trans** | 针对矩阵A的枚举类型定义 | ● CCBLAS_OP_N：表示不做任何操作<br>● CCBLAS_OP_T：表示对指定对象做转置操作 |
| **ccblasDiagType_t diag** | 指示主对角线上的元素是否是全1 | ● CCBLAS_DIAG_NON_UNIT：表示对角线有非1元素<br>● CCBLAS_DIAG_UNIT：表示对角线元素全为1 |
| **int32_t n** | 矩阵A的行数和列数 | 大于0的整数值，满足参数A的要求 |
| **int32_t K** | 矩阵的带状宽度 | 大于0的整数值，满足参数A的要求 |
| **const half_float * A** | 输入矩阵A的起始地址 | 不能为NULL<br>**说明**<br>维度为lda×n的矩阵，lda>=k+1 |
| **int32_t lda** | 用于存储矩阵A的二维数组的第一个维度值 | lda>= max(1,n) |

  

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **half_float * x** | 输入向量x的起始地址<br>同时也是输出向量的起始地址 | 不能为NULL<br>**说明**<br>x的长度>=（1+（n-1）*incx） |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 非0的整数 |

### 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回错误码。

### 相关函数

无。

## 2.2.6 ccblas_device_htpsv

### 函数功能

该函数实现了op(A)x=b方程求x的过程。

### 函数格式

**int32_t ccblas_device_htpsv(**

**ccblasFillMode_t uplo,**

**ccblasOperation_t trans,**

**ccblasDiagType_t diag,**

**int32_t n,**

**const half_float *AP,**

**half_float *x,**

**int32_t incx);**

### 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **ccblasFillMode_t uplo** | 指示三角矩阵A存储上半部分或下半部分，其他元素未引用或可从存储的部分进行推导 | ● CCBLAS_FILL_MODE_LOWER存储下半部分<br>● CCBLAS_FILL_MODE_UPPER存储上半部分 |

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **ccblasOperation_t trans** | 针对矩阵A的枚举类型定义 | ● CCBLAS_OP_N：表示不做任何操作<br>● CCBLAS_OP_T：表示对指定对象做转置操作 |
| **ccblasDiagType_t diag** | 指示主对角线上的元素是否是全1 | ● CCBLAS_DIAG_NON_UNIT：表示对角线有非1元素<br>● CCBLAS_DIAG_UNIT：表示对角线元素全为1 |
| **int32_t n** | 矩阵A的行数和列数 | 大于0的整数值，满足参数x的要求 |
| **const half_float * AP** | 输入矩阵的首地址 | 不能为NULL<br>**说明**<br>长度>=(n*(n+1))/2 |
| **half_float *x** | 输入向量x首地址<br>同时也是输出向量的首地址 | 不能为NULL<br>**说明**<br>x的长度=1+(n-1)*abs(incx) |
| **int32_t incx** | 向量x连续两个元素之间的步长 | 非0整数 |

### 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回**CCBLAS_STATUS_EXECUTION_FAILED**

### 相关函数

无。

# 2.3 Level3 接口

## 2.3.1 ccblas_device_hgemm

### 函数功能

该函数实现矩阵与矩阵乘法。

### 函数格式

**int32_t ccblas_device_hgemm(**

**ccblasOperation_t transa,**

**ccblasOperation_t transb,**

**int32_t m,**

**int32_t n,**

**int32_t k,**

**const half_float alpha,**

**const half_float* A,**

**int32_t lda,**

**const half_float* B,**

**int32_t ldb,**

**const half_float beta,**

**half_float* C, int32_t ldc);**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **ccblasOperation_t transa** | 针对矩阵A的枚举类型定义 | ● CCBLAS_OP_N：表示不做任何操作<br>● CCBLAS_OP_T：表示对指定对象做转置操作 |
| **ccblasOperation_t transb** | 针对矩阵B的枚举类型定义 | ● CCBLAS_OP_N：表示不做任何操作<br>● CCBLAS_OP_T：表示对指定对象做转置操作 |
| **int32_t m** | 矩阵A和C的行数 | 大于0的整数，满足A/C的要求 |
| **int32_t n** | 矩阵B和C的列数 | 大于0的整数，满足B/C的要求 |
| **int32_t k** | 矩阵A的列数，和B的行数。 | 大于0的整数，满足A/B的要求 |
| **half_float alpha** | 标量，比例因子 | 任意值 |
| **const half_float* A** | 输入矩阵A的起始地址 | 不能为NULL<br>**说明**<br>1.当非转置时，ldaxk的矩阵，其中lda>=max(1,m)<br>2.当转置时，A是ldaxm维的，其中lda>=max(1,k) |
| **int32_t lda** | 用于存储矩阵A的二维数组的第一个维度值 | lda>=max(1,m) |

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **const half_float* B** | 输入矩阵B的起始地址 | 不能为NULL<br>**说明**<br>1.当非转置时，ldbxn的矩阵，其中ldb>=max(1,k)<br>2.当转置时，B是ldbxk维的，其中lda>=max(1,n) |
| **int32_t ldb** | 用于存储矩阵B的二维数组的第一个维度值。 | ldb>=max(1,k) |
| **half_float beta** | 标量，比例因子 | 任意值 |
| **half_float* C** | 输入矩阵C的起始地址<br>同时也是输出矩阵C的起始地址 | 不能为NULL<br>**说明**<br>维度为ldc×n的矩阵，ldc>=max(1,m) |
| **int32_t ldc** | 用于存储矩阵C的二维数组的第一个维度值 | ldc>=max(1,m) |

## 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回错误码。

## 相关函数

无。

# 2.3.2 ccblas_device_hgemm_ex

## 函数功能

该函数实现矩阵与矩阵乘法。

## 函数格式

**int32_t ccblas_device_hgemm_ex(**

**int transa,**

**int transb,**

**int32_t m,**

**int32_t n,**

**int32_t k,**

**const __fp16 alpha,**

**const __fp16 * A,**

**int32_t lda,**

**const __fp16 * B,**

**int32_t ldb,**

**const __fp16 beta,**

**__fp16 * C,**

**int32_t ldc);**

## 参数说明

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **int transa** | 针对矩阵A的类型定义 | ● 0：表示不做任何操作<br>● 1：表示对指定对象做转置操作 |
| **int transb** | 针对矩阵B的类型定义 | ● 0：表示不做任何操作<br>● 1：表示对指定对象做转置操作 |
| **int32_t m** | 矩阵A和C的行数 | 大于0的整数，满足A/C的要求 |
| **int32_t n** | 矩阵B和C的列数 | 大于0的整数，满足B/C的要求 |
| **int32_t k** | 矩阵A的列数，和B的行数。 | 大于0的整数，满足A/B的要求 |
| **__fp16 alpha** | 标量，比例因子 | 任意值 |
| **const __fp16* A** | 输入矩阵A的起始地址 | 不能为NULL<br>**说明**<br>1.当非转置时，ldaxk的矩阵，其中lda>=max(1,m)<br>2.当转置时，A是ldaxm维的，其中lda>=max(1,k) |
| **int32_t lda** | 用于存储矩阵A的二维数组的第一个维度值 | lda>=max(1,m) |
| **const __fp16* B** | 输入矩阵B的起始地址 | 不能为NULL<br>**说明**<br>1.当非转置时，ldbxn的矩阵，其中ldb>=max(1,k)<br>2.当转置时，B是ldbxk维的，其中lda>=max(1,n) |
| **int32_t ldb** | 用于存储矩阵B的二维数组的第一个维度值。 | ldb>= max(1,k) |
| **__fp16 beta** | 标量，比例因子 | 任意值 |

  

| 参数 | 说明 | 取值范围 |
| --- | --- | --- |
| **__fp16* C** | 输入矩阵C的起始地址<br>同时也是输出矩阵C的起始地址 | 不能为NULL<br>**说明**<br>维度为ldc×n的矩阵，ldc>=max(1,m) |
| **int32_t ldc** | 用于存储矩阵C的二维数组的第一个维度值 | ldc>=max(1,m) |

## 返回值

函数执行成功时返回**CCBLAS_STATUS_SUCCESS**

函数执行失败时返回错误码。

## 相关函数

无。

# A 免责声明

- 本文档可能包含第三方信息、产品、服务、软件、组件、数据或内容（统称“第三方内容”）。华为不控制且不对第三方内容承担任何责任，包括但不限于准确性、兼容性、可靠性、可用性、合法性、适当性、性能、不侵权、更新状态等，除非本文档另有明确说明。在本文档中提及或引用任何第三方内容不代表华为对第三方内容的认可或保证。
- 用户若需要第三方许可，须通过合法途径获取第三方许可，除非本文档另有明确说明。

  

# B 如何获取华为帮助

日常维护或故障处理过程中遇到难以解决或者重大问题时，请寻求华为技术有限公司的技术支持。

## B.1 联系华为前的准备

为了更好的解决故障，建议在寻求华为技术支持前做好必要的准备工作，包括收集必要的故障信息和做好必要的调试准备。

## B.2 联系华为技术支持

故障处理过程中遇到难以确定或解决的问题时，请联系华为技术有限公司客户服务中心（电话：4008229999、网址：**http://enterprise.huawei.com**）。同时，您在向华为工程师反馈问题时，请注意收集以下信息：

- 用户名称、地址。
- 联系人姓名、电话号码。
- 故障发生的具体时间。
- 故障现象的详细描述。
- 设备类型、硬件型号及软件版本。
- 故障后已采取的措施和结果。
- 问题的级别及希望解决的时间。

**说明**

对于以上介绍的可能在本产品上出现的故障现象，按参考处理建议操作后，如果故障仍无法得到解决，请及时与就近的华为办事处或客户服务中心联系，以便能够快速获取华为公司的技术支持。

## B.3 做好必要的调试准备

在寻求华为技术支持时，华为技术支持工程师可能会协助您做一些操作，以进一步收集故障信息或者直接排除故障。

在寻求技术支持前请准备好单板和端口模块的备件、螺丝刀、螺丝、串口线、网线等可能使用到的物品。

  

# B.4 如何使用文档

华为技术有限公司提供全面的随设备发货的指导文档。指导文档能解决您在日常维护或故障处理过程中遇到的常见问题。

为了更好的解决故障，在寻求华为技术支持前，建议充分使用指导文档。

# B.5 如何从网站获得帮助

华为技术有限公司通过办事处、公司二级技术支持体系、电话技术指导、远程支持及现场技术支持等方式向用户提供及时有效的技术支持。

## 技术支持网址

查阅技术支持网站上的技术资料：

- 企业网网址：**http://e.huawei.com**
- 运营商网址：**http://carrier.huawei.com**

## 获取华为技术支持

如果在设备维护或故障处理过程中，遇到难以确定或难以解决的问题，通过文档的指导仍然不能解决，请通过如下方式获取技术支持：

- 联系华为技术有限公司客户服务中心。

  中国区企业用户请通过以下方式联系华为：
  - 客户服务电话：400-822-9999
  - 客户服务邮箱：**ChinaEnterprise_TAC@huawei.com**

    企业网全球各地区客户服务热线可以通过以下网站查找：**企业用户全球服务热线**

    中国区运营商用户请通过以下方式联系我们：
  - 客户服务电话：400-830-2118
  - 客户服务邮箱：**support@huawei.com**

    运营商全球各地区客户服务热线可以通过以下网站查找：**运营商用户全球服务热线**
- 联系华为技术有限公司驻当地办事处的技术支持人员。

## 案例库与自助平台

如果您想进一步学习和交流：

- 访问**华为服务器信息自助服务平台**，获取相关服务器产品资料。
- 访问**华为服务器自助问答系统**，快速查询产品问题。
- 访问**华为企业互动论坛（服务器）**，进行学习交流。
- 参阅已有案例进行学习：**华为服务器案例库**。

# B.6 联系华为的方法

华为技术有限公司为客户提供全方位的技术支持，用户可与就近的办事处联系，也可直接与公司总部联系。

华为技术有限公司

地址：深圳市龙岗区坂田华为总部办公楼

邮编：518129

网址：**http://enterprise.huawei.com/**