PCIe-6603

编程说明书

版权声明

目　　录

1 原理

控制器内在内核驱动程序内创建了一个实时任务，默认的运行周期为 1ms；该实时任务管理着一个大容量指令队列（默认为 16M 字节），实时任务每执行一次，从中取出 1024 条指令压入 FPGA 内部的硬件指令队列。大容量指令队列分为了 4 块，用户每次最多可写满其中的一块。

FPGA 内的指令解释器以 100MHz 的时钟运行，指令执行时间为两个时钟周期。该指令解释器支持设置振镜位置、设置数字输出、设置延时、设置 PWM 指令。

2 函数说明

2.1 前提约定

2.1.1 物理单位

下面是函数库中使用到的物理量及其解释：

名称	英文名称	单位	描述
位置	Position	P	脉冲个数
速度	Velocity	P/s, PPS	脉冲每秒
加速度	Acceleration	P/s^2	脉冲每平方秒
加速度率	Jerk	P/s^3	脉冲每立方秒
电压	voltage	V	伏特
时间	time	s	秒
时间	time	ns	纳秒, 1 s = 1000000000 ns

2.1.2 数据类型

下面是函数库中使用到的数据类型及其解释：

数据类型	范围	说明
int	-2147483647 ~ 2147483648	有符号整形，32 位
unsigned int	0~4294967295	无符号整形，32 位
double	-1.7976931348623158e+308~ 1.7976931348623158e+308	双精度浮点数，64 位
	NaN	Not A Number, FPU 产生异常后的返回值，比如负数的平方根
	Inf	infinity, FPU 产生异常后的返回值，比如被零除

2.1.3 返回值

函数库内所有函数都有返回值，用来表示函数执行过程是否出现错误。应用程序在调用函数后检查返回值，及时进行异常处理，可有效提高应用程序的健壮性。

函数返回值为 int 类型，其定义如下：

返回值	描述	原因分析
0	指令发送并执行成功	
-1	无法建立与实时任务的通讯	没有安装驱动程序
-2	发送指令超时	驱动程序和函数库版本不匹配
-3	无法执行指令	执行指令的前置条件不满足
-4	读取状态超时或数据不完整	驱动程序和函数库版本不匹配
-5	无效参数	指令参数无效
-6	设备已经打开	设备已经被应用程序打开
-7	分配系统资源失败	操作系统资源严重不足

2.1.4 打开与关闭

控制命令	int LC_Open(void)
功能描述	查找并打开系统里安装的控制卡
参数	无
返回	参考返回值
前置条件	无
后置条件	无

控制命令	int LC_Close (void)
功能描述	关闭已经打开的控制卡并释放资源
参数	无
返回	参考返回值
前置条件	无
后置条件	无

2.1.5 缓冲区管理

控制命令	int LC_SubmitBuffer(int dev, unsigned int* buffer, unsigned int count, unsigned int* act)
功能描述	向驱动程序发送一块指令，指令定义参考后面表格。 发送前需要调用 LC_GetStatus 函数查询控制器内可用的指令队列大小(buffer[6])，发送的指令大小不得超过可用的指令队列大小（指令数 count 不得超过 buffer[6]/4）。 当 buffer[6]为 0 时，表示指令队列满，暂时不能接收新数据。
参数	dev – 设备号 buffer – 指令缓冲区 count – 指令缓冲区内指令数目 act – 实际写入的指令数目
返回	参考返回值
前置条件	无
后置条件	无

指令说明表格：

[31:28]	[27:24]	[23:16]	[15:8]	[7:0]	说明
0x4			position		暂存 X 位置
0x5			position		保存 Y 位置并将之前暂存的 Z,X 位置一起发送到 XY2-100 控制器
0x6			position		暂存 Z 位置
0x7	delay				延时，范围 0~1048575，单位为 10us
0x8		port value			设置 TTL 类型的数字输出(DO0~DO23)，设置输出端口，对应位为 1 表示输出高电平，0 表示低电平
0x9	value				设置模拟量输出（保留）
0xa	value			bit	设置一个 TTL 类型的数字输出位(DO0~DO23)，其中 bit[24]表示输出状态(1 表示输出高电平，0 表示低电平)，[4:0]表示位号
0xb			width		暂存 PWM 脉冲宽度参数，单位为 40ns
0xc			period		保存 PWM 周期参数并将之前保存的脉冲宽度参数一起设置到 PWM 控制器

C/C++里的指令定义如下：

//command accepted by built-in XY2-100 controller

#define XY2_CMD_POS_X(d) ((0x4u << 28) | ((d) & 0xffffff)) //store X pos

#define XY2_CMD_POS_Y(d) ((0x5u << 28) | ((d) & 0xffffff)) //store Y pos and then send XYZ to shift register

#define XY2_CMD_POS_Z(d) ((0x6u << 28) | ((d) & 0xffffff)) //store Z pos

#define XY2_CMD_DELAY(d) ((0x7u << 28) | ((d) & 0xffffff)) //0~1048575, delay unit in 10-microsecond,

#define XY2_CMD_DOUT(d) ((0x8u << 28) | ((d) & 0xffffff)) //set dout port

#define XY2_CMD_AOUT(d) ((0x9u << 28) | ((d) & 0xffffff)) //set analog output(reserved for future)

#define XY2_CMD_DOUT_BIT(d,n) ((0xau << 28) | (((d) & 0x1) << 24) | (n & 0x1f)) //set a dout bit

#define XY2_CMD_PWMW(d) ((0xbu << 28) | ((d) & 0xffffff)) //store PWM width in 40-nanosecond

#define XY2_CMD_PWMP(d) ((0xcu << 28) | ((d) & 0xffffff)) //store PWM period in 40-nanosecond and then set PWM controller

2.1.6 状态查询

控制命令	int LC_GetStatus(int dev, unsigned int* buffer, unsigned int count)
功能描述	查询控制器状态 在缓冲区足够大情况下，返回的状态定义如下： buffer[0]-数字输入端口状态（bit0~bit21 分别对应 DI0~DI21）；对于光耦输入 1 表示导通，0 表示截止；对于 TTL 输入 1 表示高电平(>2.0V)，0 表示低电平(<0.8V)； buffer[1]-光耦隔离的数字输出端口状态（bit24~bit31 分别对应 DO24~DO31）；对于光耦输出 1 表示导通，0 表示截止；对于 TTL 输出 1 表示高电平，0 表示低电平； buffer[2]-振镜状态状态，[15:0]为状态字节,[18:16]为主题指示； buffer[3]-XY2-100 控制器错误标记，1 表示运行过程中出现队列空，0 表示无错误； buffer[4]-控制器内 PLL 错误标记，1 表示 PLL 出现错误； buffer[5]-驱动程序管理的指令队列大小，单位为字节 buffer[6]-单次可写入的指令大小，单位为字节；队列满时此值为 0；该标记可作为是否可写入指令的标记，一次可写入的指令数为 buffer[6]/4； buffer[7]-指令队列读位置 buffer[8]- 指令队列写位置 buffer[9]-指令队列内已存的指令块数目，范围为 0~4；0 表示队列为空，4 表示队列满； buffer[10]- 指令队列满标记，1 表示队列满，0 表示队列未满。

	buffer[11]- 实时任务周期最小值，单位 ns buffer[12]- 实时任务周期最大值，单位 ns buffer[13]- 实时任务周期平均值，每 1024 个周期求平均一次，单位 ns 注：灰色内容表示可能会更改或删除的定义。
参数	dev – 设备号 buffer – 状态缓冲区 count – 缓冲区元素数目
返回	参考返回值
前置条件	无
后置条件	无

2.1.7 停止

控制命令	int LC_Abort(int dev)
功能描述	停止发送数据 收到此命令后，实时任务将其管理的指令队列清空，同时将状态信息内的 xy2_pll_error、xy2_starvation_err 清零。 注意该命令仅仅清除实时任务管理的大容量指令队列（默认为 16M 字节），而 FPGA 内部的硬件指令队列（仅使用了 1024 级）则不处理。
参数	dev – 设备号
返回	参考返回值
前置条件	无
后置条件	无

2.1.8 设置数字量输出

控制命令	int LC_SetDout(int dev, int index, int value)
功能描述	设置光耦隔离的数字量输出(DO24~DO31)； TTL 类型的数字量输出(DO0~DO23)使用缓冲区指令来设置。
参数	dev – 设备号 index – 输出编号，24~31 value – 数字量输出值，1 表示导通，0 表示截止。
返回	参考返回值

前置条件	无
后置条件	无

2.1.9 控制器信息查询

控制命令	int LC_GetDevInfo(int dev, unsigned int* buffer, unsigned int count)
功能描述	查询控制器的状态信息
参数	buffer - 信息缓冲区 count - 缓冲区元素个数 在缓冲区足够情况下，查询到的状态信息定义如下： buffer [0] - 控制器 ID 号 buffer [1] - 控制器控制轴数目 buffer[2] - 控制器数字量输入输出数目 buffer[3] - 控制器模拟量输出数目 buffer[4] - 控制器 FPGA 版本号 buffer[5] - 控制器驱动程序版本号 buffer[6] - 控制器函数库版本号 buffer[7] - 控制器 BOM 编号 buffer[8] – 保留 buffer[9] - 控制器实时任务运行周期，单位 ns
返回	参考**错误!未找到引用源。**
前置条件	无
后置条件	无

3 使用流程

3.1 发送数据

1) 使用 LC_GetStatus 函数查询控制器内部的状态信息，如果指令队列存在未用空间，buffer[6]返回可用的大小(可一次写入的指令大小,换算为指令数时需要除 4)；如果指令队列已满,buffer[6]返回 0；

注意：只有在函数值返回值为 0 的情况下，buffer 内的状态信息才是有意义的；

2) 使用 LC_SubmitBuffer 函数发送指令到控制器，指令数不得超过 buffer[6]/4（buffer 为上一步使用 LC_GetStatus 函数查询到的状态信息)；注意检查 LC_SubmitBuffer 函数返回值；控制器实际接收到的指令数可通过 act 参数查询到。每次调用 LC_SubmitBuffer 函数之前都需要调用 LC_GetStatus 函数来查询可用的空间大小。

3) 用户可重复上述 1、2 步骤，直到把所有数据发送到控制器的指令队列内。为了提高效率，避免指令队列出现空情况，一次发送的指令数要尽可能多（尽量接近 buffer[6]/4)。

4) 指令发送完毕后，额外发送不少于 1024 个延时指令（延时可设置为 0，或者其它允许的指令)，然后调用 LC_GetStatus 函数查询控制器状态，如果 buffer[9]为 0，表明所有的用户指令已经发送完毕。控制器内的实时任务管理着大容量指令队列（默认为 16M 字节)，实时任务每执行一次，从中取出 1024 条指令压入 FPGA 内部的硬件指令队列。此步骤的目的是保证所有的用户指令已经被硬件执行完毕。

3.2 异常处理

用户程序在执行的过程中，需要通过调用 LC_GetStatus 函数得到的状态信息来判断是否出现异常：如果 buffer[3]不等于 0，表明在执行的过程中 FPGA 内的硬件指令队列出现了空情况（控制器内的实时任务管理的大容量队列内仍有数据)，此时激光标刻的轨迹可能出现断续的情况，此时需要人工干预处理。 buffer[4]等于 1，表明出现硬件错误，控制器内的 PLL 出现失去锁定错误。

调用 LC_Abort 函数后，控制器内的指令队列空错误、PLL 失去锁定错误可被清零。

4 修订记录

日期	版本	修改说明
2020-9-10	1.00	创建