激光API测试文档

功能API

1. 用于获取操作的结果, 返回Int整形结果。

/// <summary>

/// 获取signal对应操作的结果。在setSignal返回后，调用此API

/// </summary>

/// <returns></returns>

int signal.GetSignalOperationIntResult();

string signal.GetSignalOperationStringResult();

1. AIV 小车充电控制

2.1 /// <summary>

/// 请求充电 -- 设置aiv的继电器连接上

/// </summary>

/// <returns></returns>

bool aiv\_charge.ChargeReady();

2.2 /// <summary>

/// 充电完成--设置aiv断开内部继电器。

/// </summary>

/// <returns></returns>

bool aiv\_charge.ChargeFinished();

2.3 /// <summary>

///

/// </summary>

/// <returns></returns>

bool aiv\_battery.IsLowPower()

2.4 /// <summary>

/// 获取当前点的充电桩序号 -- 根据当前的workpoint id。

/// </summary>

/// <param name="workPointId"></param>

/// <returns></returns>

int aiv\_battery.GetChargingDeviceID(int workPointId);

2.5//充电桩

//chargingMsgType = 1; // 充电命令类型值

//chargeReqMsgId = 2; // 请求充电桩伸出充电头的消息ID

//chargeFinishMsgId = 3; // 请求充电桩收回充电头的消息ID

//string signal.GetSignalName(int signalType, int nSignalId, int nSN);

//---- signal 操作

signal.SetSignal(signalName);

signal.WaitSignal(signalName);

2.6请求充电API

/// <summary>

/// 请求充电

/// </summary>

/// <param name="powerCap">电池电量</param>

/// <returns></returns>

task.RequestCharging(double powerCap);

TASK

3.1. 关于aiv参与工作的操作通知API：

/// <summary>

/// 2017-06-14 add an API

/// task service need implement this API.

/// Online Service Component doesn't need implement this API.

/// </summary>

/// <param name="operationSignal"></param>

/// <returns></returns>

bool StartOperation(string operationSignal);

bool FinishOperation(string operationSignal);

3.2获取下一工作点许可

PointGrantStatus ITaskDispatchService.GetNextTaskPointPermission(int taskPointId)

void GetNextTaskPointPermissionUntillGrant(int taskPoint);

PointGrantStatus: {

Refused = 0,

Granted = 1,

Pending = 2,

WAITING ,

}

sample:

// 直到granted 才返回。

GetNextTaskPointPermissionUntillGrant(nextPointId);

PointGrantStatus = taskresult.ResultOnPointPermission()

or:

nextPermission = task.GetNextTaskPointPermission(nextPointId);

// 直到granted（1）

while (nextPermission ~= 1) do

aiv.WaitTimeout(1000);

nextPermission = taskresult.ResultOnPointPermission()

end

3.3.aiv.EnableOriginPoint(bool)

3.4 .laser设定安全模式的api：

laser.SetSecurityLaserMode(int nMode)

RFID

4.1.获取当前RFID

string RFID.GetCurrentRFID()

4.2. 更新设置直行RFID API:

RFID.SetStraightControlRFID("rfid")

4.3. 添加设置通过下个rfid后的速度

RFID.SetNextRFIDSpeed("rfid",speed);

\*\* 使用时，需要PLC 程序更新匹配。

4.4等待读取RFID id为等待读取编号

id1 = "0020"

bResult=RFID.WaitRFID(id1, fal se)

4.5设置RFID控制API

示例：

RFID.SetStopControlRFID("0308");

RFID.SetLeftControlRFID("0402");

RFID.SetRightControlRFID("2944");

动作API

5.1PLC控制转角的API

aiv.SetTurnThetaCommand(theta);

5.2完成磁导的divergence路径选择api：

/// <summary>

/// 磁条导航中转左路径

/// </summary>

void TurnToLeftPathByPermeameter();

5.3

/// <summary>

/// 磁条导航中转右路径

/// </summary>

void TurnToRightPathByPermeameter();

5.4

/// <summary>

/// 磁条分叉路口中直行路径

/// </summary>

void TurnStraightPathByParmeameter();

5.5 停止API：void Stop();

5.6. aiv.MoveWithSpeedByPermeameter(800, true)

5.7 aiv.TurnThetaWithWheel(double theta, bool turnLeft);

5.8. void TurnTheta(double theta,double w,bool leftTurn); // 使用时，请先关闭磁导模式。

5.9. void MoveToPointAbsolute(double s, double x, double y, double theta, double maxVelocity);