

A man in a light blue shirt is seen from the side, looking at a tablet. The background is a blurred industrial factory floor with overhead lights and machinery. Overlaid on the image are several digital graphics: a Siemens logo in the top right, a '24/7' circular icon, a 'NEWS' section with a person icon, a 'Home' button, and a network diagram with three people icons. The text 'Industry Online Support' is also visible.

SIEMENS

SIMATIC S7-200 SMART 基于位置的队列调度库

STEP 7-Micro/WIN SMART V2.8

法律信息

应用实例的使用

应用实例说明了通过文本、图形和/或软件模块形式的几个组件的交互来实现自动化任务的解决方案。本应用程序示例是由西门子公司和/或西门子公司(以下简称“西门子”)的子公司提供的免费服务。它们是非约束性的,并且不声明关于配置和设备的完整性或功能性。应用程序示例仅提供典型任务的帮助;它们并不构成客户特定的解决方案。您有责任按照适用的法规,对产品的正确和安全操作负责,并必须检查相应的应用示例的功能,并为您的系统定制它。

西门子授予您非排他性、不可再授权和不可转让的权利,让经过技术培训的人员使用应用示例。

对应用程序示例的任何更改都由您负责。与第三方共享应用示例,或复制应用示例或摘录,仅允许与您自己的产品结合使用。该应用实例无须接受收费产品的惯常测试和品质检验;它们可能有功能和性能缺陷以及错误。您有责任使用它们,使任何可能发生的故障不会导致财产损失或人身伤害。

免责声明

由于任何法律原因, Siemens 不承担任何责任,包括但不限于对应用示例的可用性、完整性和不存在缺陷以及相关信息、配置和性能数据以及由此造成的任何损害承担责任。这个不适用强制责任的情况下,例如在德国的产品责任法,或意图的情况下,重大过失,或有罪的生命损失,人身伤害或损坏健康,不符合担保,欺骗性的非披露缺陷或有罪的违反合同义务。但因违反重大合同义务而提出的损害赔偿要求应限于协议类型的典型可预见损害,但因故意或重大过失或基于生命损失、身体伤害或健康损害而产生的责任除外。上述规定并不意味着对您不利的举证责任的任何改变。对于第三方在此方面的现有或未来索赔,您应向西门子作出赔偿,除非西门子负有强制责任。

通过使用应用示例,您承认西门子对上述责任条款之外的任何损害不承担责任。

其他信息

西门子保留随时更改应用示例的权利,无需另行通知。如果应用实例中的建议与其他西门子出版物(如目录)之间存在差异,则应优先考虑其他文件的内容。

安全信息

西门子提供具有工业安全功能的产品和解决方案,支持工厂、系统、机器和网络的安全运行。

为了保护工厂、系统、机器和网络免受网络威胁,有必要实施——并持续维护——一个整体的、最先进的工业安全概念。西门子的产品和解决方案构成了这一概念的一个元素。

客户有责任防止对其工厂、系统、机器和网络未经授权的访问。

这些系统、机器和组件只应在必要的情况下连接到企业网络或 Internet,并且只有在适当的安全措施(例如防火墙和/或网络分割)到位的情况下才应连接到这种连接。有关可能实施的工业保安措施的其他资料,请浏览 <https://www.siemens.com/industrialsecurity>。

西门子的产品和解决方案经过不断的发展,使其更加安全。西门子强烈建议,一旦产品更新可用,就立即应用产品更新,并使用最新的产品版本。使用不再受支持的产品版本以及未能应用最新更新可能会增加客户遭受网络威胁的风险。

了解产品更新,请订阅西门子工业安全 RSS Feed: <https://www.siemens.com/industrialsecurity>。

目录

- 1 应用概述..... 4
 - 1.1 通用描述 4
 - 1.2 硬件及软件需求 4
- 2 程序说明..... 5
 - 2.1 简要说明 5
 - 2.2 使用说明 7
- 3 更新日志..... 11

1 应用概述

1.1 通用描述

SIMATIC S7-200 SMART 用于流水线机型中时，经常会需要基于编码器数值进行输送带上各工件进行位置比对，并输出信号至各工位对工件进行加工的情况；以图中所示流水线模型为例，各工位需在工件到达设定位置时启动，并在工件离开设定位置时停止，当工件长度与间隔不一致时，则需要对各个工件的头尾位置与尾部位置进行实时计算，此举会占用大量的寄存器空间以及运算资源，并会伴随一定的编程难度，因此开发此库以便于在类似情况时快速开发出应用且占用较小的运算资源。

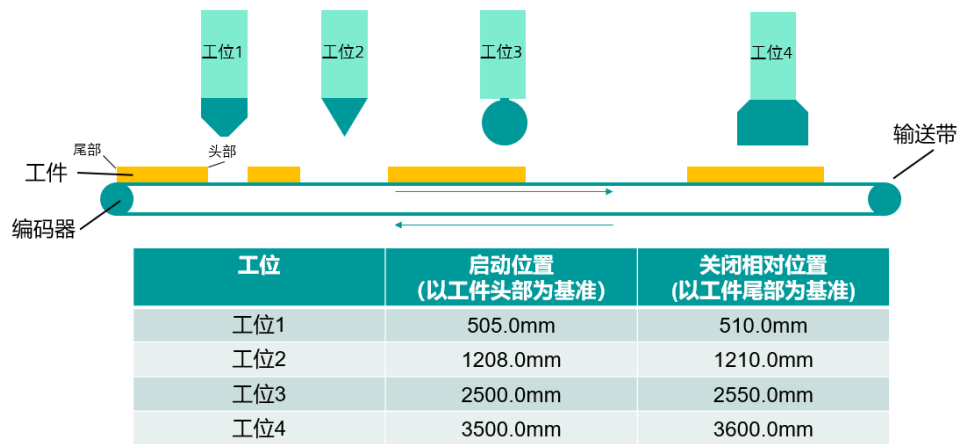


图 1.1 流水线模型

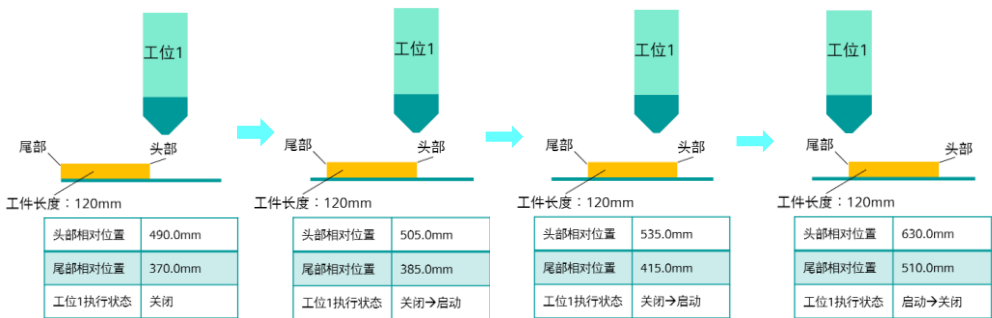


图 1.2 工位启动时序与工件位置关系

1.2 硬件及软件需求

本应用软硬件的需求

硬件

- ST20/ST30/ST40/ST60/SR20/SR30/SR40/SR60 固件版本 V2.8
- 输入的位置坐标需是单向递增，可不处理溢出，但不允许有反向

软件

- STEP 7-Micro/WIN SMART V2.8

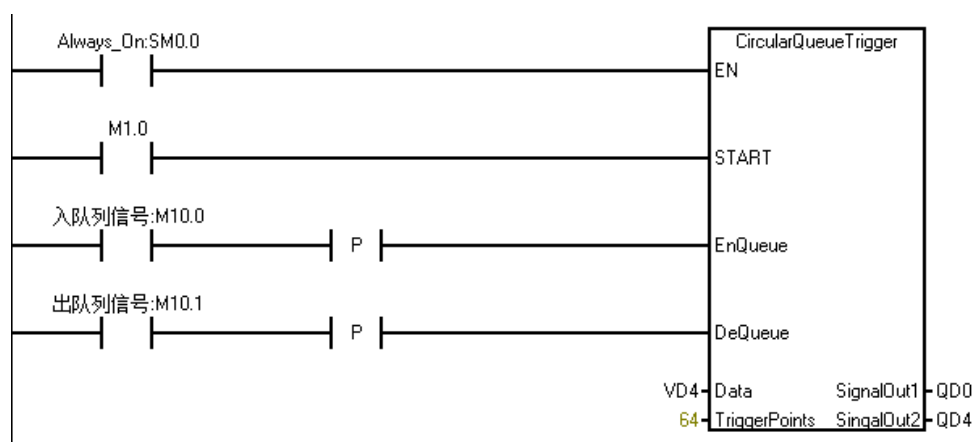
2 程序说明

2.1 简要说明

该库主要包含以下功能：

- 1、位置数据的入队列/出队列。
- 2、按入队列的顺序对队列内位置数据进行比对并触发信号，最大可触发信号数量为 64 个。
- 3、各由用户设定的触发位置存于在库占用存储区中的以“QCT_TrPos1”变量为起始的 64 个 Dint 数据中。
- 4、CircularQueueTrigger2 与 CircularQueueTrigger 功能一致，为便于工件长度不一致时记录同一工件的不同位置（如 CircularQueueTrigger 用于记录工件前部位置，CircularQueueTrigger2 用于记录工件尾部位置）。

编码功能块



编码程序块引脚

参数 & 类型	数据类型	描述
EN	BOOL	程序块使能
START	IN BOOL	启动信号，运行时保持，上升沿时初始化
EnQueue	IN BOOL	入队列信号，输入上升沿脉冲有效，将当前坐标数值存入队列中
DeQueue	IN BOOL	出队列信号，输入上升沿脉冲有效，最先进坐标从队列中删除

2 程序说明

Data	IN	DINT	当前坐标数值
TriggerPoints	IN	INT	触发点数
SignalOut1	OUT	DWORD	触发信号输出 1
SignalOut2	OUT	DWORD	触发信号输出 2

表 1、程序块引脚定义

库占用存储区中符号定义

参数 &同类参数数量		数据类型	描述
QCT_Index	1	DINT	库数据队列中有效数据的起始位置，根据 DeQueue 信号递增，仅读取
QCT_Length	1	DINT	库数据队列中有效数据的长度，范围 1-300，仅读取
QCT_Started	1	BOOL	库启动信号锁存，内部使用，仅读取
QCT_Cycle	1	WORD	库需判断触发位置的数量，内部使用，仅读取
QCT_Data1	300	DINT	库数据队列存储区，内部使用，仅读取
QCT_Data300			
QCT_TrPos1	64	DINT	库用于判断的触发位置，必须由用户写入，触发位置需连续存储，可读写
QCT_TrAdr1	64	DINT	库用于指向需判断的数据地址的触发地址指针，每个触发位置对应一个触发地址指针，存放地址指针数据，内部使用，仅读取
QCT_TrAdr64			

2.2 库实现简介

本库各变量的含义如图

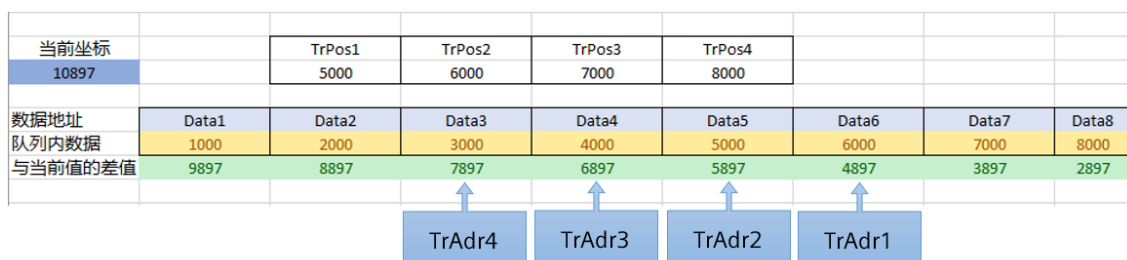


图 2.2.1 本库各变量含义

其中，TrPos1 到 TrPos4 为用户输入设定的触发位置，Data1-Data8 为由“EnQueue”触发的进入队列中的数据；TrAdr1-TrAdr4 对应储存 TrPos1 到 TrPos4 与队列中进行比对的元素的地址。

程序运行时：

数据入队列：若 START 引脚输入 ON，则当“EnQueue”为 True 时，将 Data 对应的数值保存到队列数据地址中。

数据出队列：若 START 引脚输入 ON，且当“DeQueue”为 True 时，将队列中的首元素改为 0，且将第二个元素定义为新的首元素。

数据比对：以 TrPos1 为例，START 上升沿时，将队列首地址传入 TrAdr1 中；之后在每个扫描周期中，均比对 TrAdr1 指向的元素与当前值的差值，当差值大于 TrPos1 时，输出一个周期的信号并将队列中下一个元素的地址写入 TrAdr1 中，后续扫描周期中持续比对。发生该动作时的寄存器变化如图

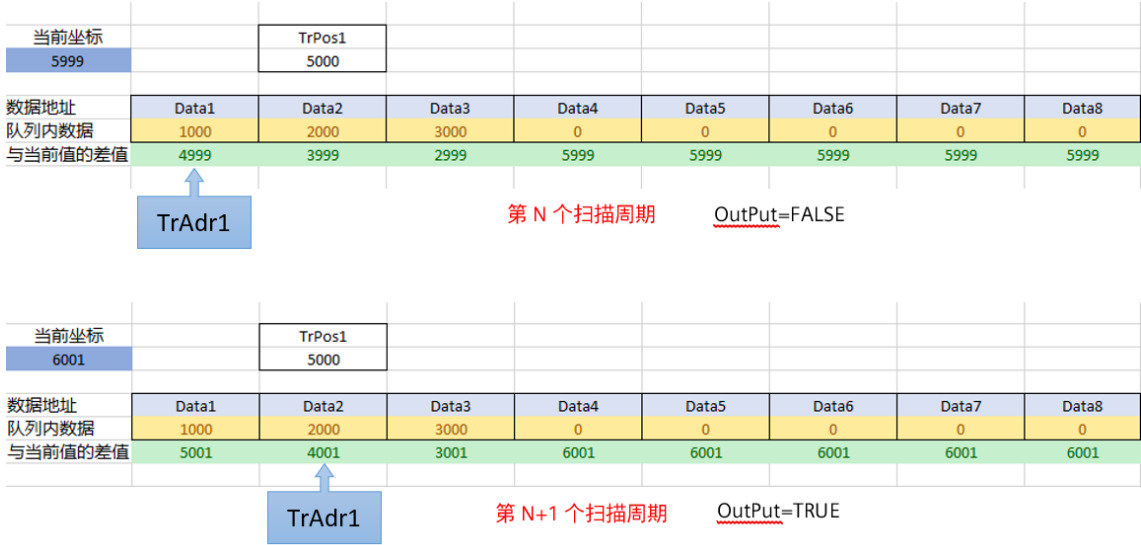


图 2.2.2 比对目标值大于设定值时的寄存器时序

2.3 库执行时序详解

START 信号初次置位时，库对所有使用到的寄存器进行复位，复位后库中各寄存器的状态如图 2.3.1 所示，此时数据队列中所有数据清零，CQT_Length 与 CQT_Index 均为 0，所有（TrAdr1-64）指向第一个数据地址。



图 2.3.1 START 置位后第一个扫描周期动作

当 EnQueue 信号首个上升沿触发时，库将当前坐标存入数据队列中，此时库中各寄存器状态如图 2.3.2 与 2.3.3 所示，此时 CQT_Length 为 1，CQT_Index 为 0，库开

始计算数据队列中触发地址指针指向的数据地址中的值与当前值的差值，该差值与触发位置进行比对。



图 2.3.2 EnQueue 上升沿输入周期时动作

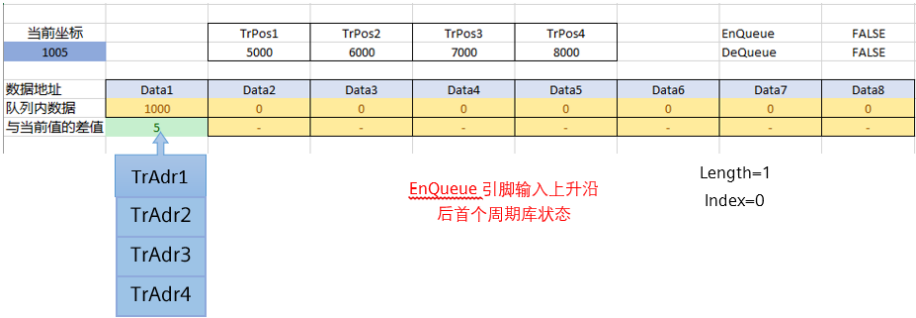


图 2.3.3 EnQueue 上升沿输入后首个周期动作

当 EnQueue 信号再次触发时，将当前坐标存入数据队列的下一个元素中（Data2），此时库中各寄存器状态如图 2.3.4 所示，此时 CQT_Length 为 2，CQT_Index 为 0。



图 2.3.4 EnQueue 上升沿再次输入时动作

当触发地址指向的数据地址中的值与当前值的差值大于对应的触发位置时，输出对应的信号并将触发地址指针指向下一个数据地址。触发时与上一周期库动作如图 2.3.5 和 2.3.6 所示



图 2.3.5 TrPos1 触发前周期动作

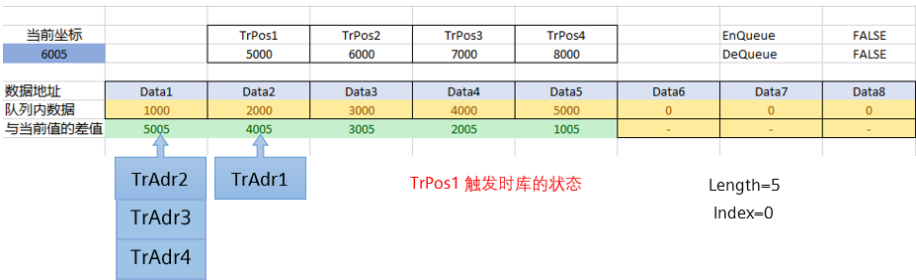


图 2.3.6 TrPos1 触发周期动作

当 DeQueue 信号上升沿触发时，库将最先进入队列的元素定义为无效数据，数据清零，所有指向该数据地址的触发地址指针均指向下一个数据地址，库动作如图 2.3.7 所示，此时 CQT_Length 为 4，CQT_Index 为 1。

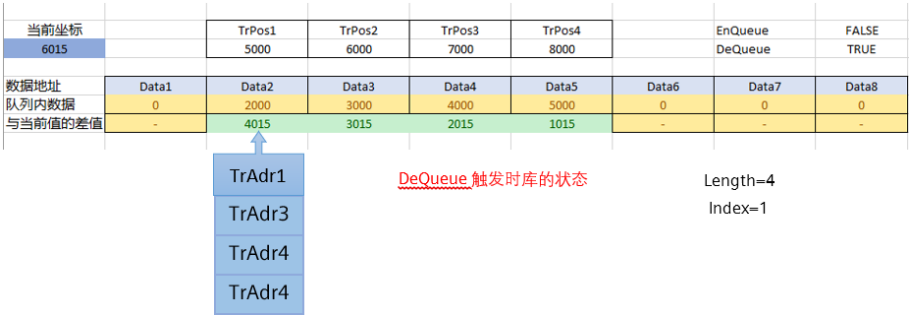


图 2.3.7 DeQueue 上升沿周期动作

2.4 使用说明

关于触发坐标的存储位置：触发坐标存于以库数据表变量“QCT_TrPos1”对应坐标为起始地址的寄存器区域中，数据类型为 DINT，最多可存储 64 个触发坐标，坐标需连续存储。

关于触发信号地址：以 SignalOut1 引脚地址为 VD0 为例，第一个触发坐标对应的触发信号点为 V3.0，第八个触发坐标对应的触发信号点为 V3.7，第九个触发坐标对应的触发信号点为 V2.0，以此类推。

关于入队列信号与出队列信号：此两信号均需要输入沿信号，若输入电平信号会导致意料之外的状态，“DeQueue”所需信号以库数据表变量“QCT_Length”不大于 300 为原则或其他判断依据给出。

关于“TriggerPoints”引脚：该引脚为需要判断的触发点数，运行时以“QCT_TrPos1”对应坐标为起始地址的“TriggerPoints”个数据作为触发坐标进行判断触发。

关于程序运行效率：

- 当触发点数为 0 时，即只执行入队列/出队列操作时，CircularQueueTigger 所需平均扫描用时为 0.15ms
- 当触发点数不为 0 时，每增加一个触发点，所需的平均扫描用时增加 0.024ms
- 调用两个队列时，所耗费的扫描时间叠加。
- START 置位后首个周期，由于存在初始化动作，扫描周期增加约 4ms。

3 更新日志

版本& 日期	更新描述
V1.0.0 01/2024	