EtherCAT 从站开发指南

发布版本：V1.5

发布日期：2023.8.11

商标声明

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，公司对本文档内容不做任何明示或暗示的声明或保证。

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

目录

一、EtherCAT 简介

前言

本文档介绍 EtherCAT 从站二次开发的简介与要求、如何搭建二次开发环境，以及 API 函数、用例程序和常见问题处理方法。为用户对 EtherCAT 进行二次开发，提供参考与说明。

读者对象

本文档主要适用于以下工程师：

- 软件工程师
- 系统测试工程师
- FPGA 工程师

背景

EtherCAT （**以太网控制自动化技术**）是一个开放架构，基于以太网为基础的现场总线系统，其名称的 CAT 为控制自动化技术（Control Automation Technology）字首的缩写。EtherCAT 是确定性的工业以太网，最早是由德国的 Beckhoff 公司研发。它于 2003 年被引入市场，于 2007 年成为国际标准，并于 2014 年成为中国国家标准。EtherCAT 的出现为系统的实时性能和拓扑的灵活性树立了新的标准。

自动化对通讯一般会要求较短的资料更新时间（或称为周期时间）、资料同步时的通讯抖动量低，而且硬件的成本要低，EtherCAT 开发的目的就是让以太网可以运用在自动化应用中。

概述

在工业自动化领域，伺服运动控制系统作为机器人及数控设备的核心组成部分，对其相关技术的研究尤为重要。目前各领域技术的进步，促使工业设备需求不断提高，传统伺服控制系统中使用的工业现场总线受限于老旧的协议技术及硬件规格，导致其数据传输速度、实时性等方面越来越难以满足逐渐提高的系统总线数据传输需求。

因此开发一种更加优良、性能更加符合标准的工业通信技术成为了自动化通信领域的人们迫切需求。

在众多通信技术中，曾用于计算机通信领域的以太网技术引起了人们的注意，Ethernet 拥有易于连接，连通方便的优点，并且价格也较为理想，传输速度可以达到工业控制所需要的标准，这些优点使它成为了用于开发工业网络的首选。通用的工业以太网定义为: 技术上与商用以太网兼容，但在产品设计上必须满足工业现场对实时性、可靠性、可互操作性、抗干扰性、本质安全性、环境适应性等方面的需要，是继现场总线之后发展起来的、被广泛认同为颇具发展前景的一种工业通信网络。工业以太网采用 TCP /IP 协议，和 IEEE 802.3 标准兼容，但在应用层会加入各自特有的协议(通常为 IEEE 802.3/IEEE 802.3u) [2]。

相对于 IT 和办公应用中的硬件成本而言，工业自动化的硬件成本更加重要。标准以太网网络几乎无法满足以上需求的现场级应用。如果每个节点使用一个独立的 以太网报文传输几个字节的周期性过程数据，那么有效数据利用率会明显下降。因为以太 网报文的最短长度为 84 字节（包括帧间距），其中的 46 个字节可以用于过程数据。例如，一个驱动器发送 4 字节的实际位置和状态信息过程数据，同时接收 4 字节的目标位置和控制字信息数据，则发送/接收报文的有效数据利用率下降到 4.8%（4/84）。另外，驱动器通常在接收到目标值后触发传输实际值需要一定的响应时间。最终，100 Mbit/s 的带宽所剩无几。

而在 IT 领域通常使用的路由（IP）和连接（TCP）协议栈需要为每个节点使用附加的协议头，会产生进一步的延时。

与商用以太网相比，工业以太网的技术特征体现在对以下方面的特殊要求：

①具有高实时性与良好的时间确定性。

②传送信息多为短帧信息，且信息交换频繁。

③容错能力强，可靠性、安全性好。

④控制网络结构具有高度分散性。

⑤控制网络协议简单、实用，工作效率高。

⑥控制设备的智能化与控制功能的自治性。

⑦与信息网络之间有高效率的通信，易于实现与 信息网络的集成。

⑧设备的可靠性与环境适应性。

⑨远距离传输。

⑩总线供电。

工业以太网有三种实现方式：TCP/IP 方式、以太网方式、修改以太网方式。

其中，TCP/IP 的方式仍然采用传统的 TCP/IP 协议栈进行通信，通过上层合理调度减少数据传输过程中的不确定性，使用这种方式有 Mobus/TCP 和 Ethernet/IP 等协议这种方式的数据传输实时性不高；以太网的方式采用标准的以太网设备，传输普通的以太网数据仍然可以使用 TCP/IP 协议，而用于传输控制信号的过程数据则使用专门的协议传输，使用这种方式的有 Ethernet Powerlink、PROFINet RT 和中国的 EPA（Ethernet for Plant Automation）等协议，可以实现较高的实时性；修改以太网的方式采用经过修改的以太网协议传输数据，而使用专门的硬件处理数据，使得响应时间小于 1ms，它的实时数据和非实时数据也是分开传输的，彻底避免数据报文冲突，使用这种方式的有 SERCOS-III、PROFINetIRT 和 EtherCAT 等协议，

EtherCAT 技术是由德国 BECKHOFF 自动化公司提出并实现的工业以太网技术，是目前最快的工业以太网解决方案。

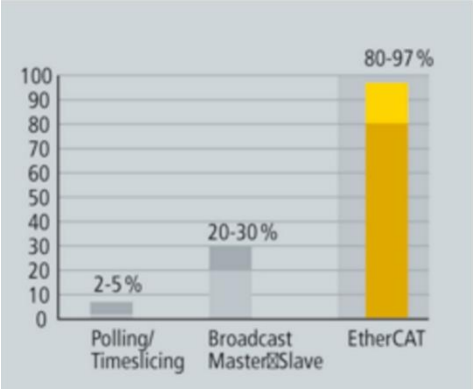


图 2

一个完整的 EtherCAT 系统可以分为**主站**和**从站**两个部分，主站使用标准以太网设备传输数据，从站使用专用的控制芯片处理数据，支持线形、树形或星形等多种拓扑结构，具有超高的性能、灵活性和成本优势，EtherCAT 协议与普通的现场总线协议相比有以下特点：

（1）EtherCAT 具有广泛的适应性，无论是简单的 16 位微处理器还是复杂的 PC 系统，只要控制单元带有普通的以太网控制器都可以构成 EtherCAT 主站。

（2）EtherCAT 是一种特殊的以太网协议，使用该协议的数据可以使用常用的以太网设备，可以节省设备更新的成本。

（3）EtherCAT 可以灵活选择从站类型，无论是带有微处理器的复杂节 点还是只有 2 位 I/O 的简单节点可以用作 EtherCAT 从站。

（4）EtherCAT 数据传输速率高，由（2）知该协议数据符合标准的以太网协议，可以通过标准的以太网进行数据传输，可以充分利用以太网带宽 进行用户数据的传输。

（5）EtherCAT 数据刷新周期短，使用专门的硬件处理数据，数据的刷新周期很小，低于 100μs，可用于响应要求高的设备。

（6） EtherCAT 具有良好的同步性能，设备之间的同步由专用硬件的相关寄存器完成，各个从站设备之间的时钟同步精度可以控制在 1μs 以内。

于是以 EtherCAT 为典型的基于 Ethernet 面向工业传输的各种各样的工业以太网技术被研发出来，是当今最流行的工业以太网标准之一。

EtherCAT 协议原理

一个 EtherCAT 数据帧足以完成所有节点控制数据的发送和接收，这种高性能的运行模 式克服了前面章节描述的各种问题。EtherCAT 主站发送一个报文，报文经过所有节点。EtherCAT 从站设备高速动态地（on the fly）读取寻址到该节点的数据，并在数据帧继续传输的同时插入数据。这样，数据帧的 传输延时只取决于硬件传输延时。当某一网段或分支上的最后一个节点检测到开放端口（无下一个从站）时，利用以太网技术的全双工特性，将报文返回给主站。EtherCAT 报文的最大有效数据利用率达 90% 以上，而由于采用全双工特性，有效数据利用率理论上高于 100 MBit/s。EtherCAT 主站是网段内唯一能够主动发送 EtherCAT 数据帧的节点，其他节点仅传送数据帧。这一设想是为了避免不可预知的延时，从而保证 EtherCAT 的实时性能 EtherCAT 主站采用标准的以太网介质访问控制器（MAC），无需额外的通信处理器。 因此，任何集成了以太网接口的硬件平台都可以实现 EtherCAT 主站，而与所使用的实时操作系统或应用软件无关。 EtherCAT 从站设备的 EtherCAT 从站控制器（ESC）负责在硬件中高速动态地（on the fly） 处理 EtherCAT 数据帧，不仅使网络性能可预测，而且其性能独立于具体的从站设备实施
方式。

EtherCAT 将其报文嵌入到标准的以太网数据帧中（形成 EtherCAT 数据帧）。设备通过 帧类型 0x88A4 识别 EtherCAT 数据帧。由于 EtherCAT 协议被优化为适用于短周期性的过程数 据，因此无需庞大的协议堆栈，例如 TCP/IP 或 UDP/IP。

为了保证节点之间的 IT 通信，TCP/IP 可选择性地通过邮箱通道传输，从而不影响实时数据的传输。在启动期间，EtherCAT 主站设备为从站设备配置并

映射过程数据。主站与从站之间交换的数据量可以各不相同，从一个位到几个字节，甚至是几 KB。 EtherCAT 数据帧包含一个或多个 EtherCAT 子报文，子报文头标明了主站设备的访问

方式：

读，写，或读-写

通过直接寻址访问指定的从站设备，或通过逻辑寻址访问多个从站设备（隐式寻址）逻辑寻址方式主要用于周期性交换的过程数据。每个报文定位到 EtherCAT 网段中过程映像的具体位置，过程映像具有 4GB 的地址空间。网络启动阶段，在全局地址空间中，为每个从站分配一个或多个地址。如果多个从站设备被分配到了相同的地址域，那么可通过单个报文对其进行寻址。由于报文中包含了所有的数据访问相关信息，因此主站可决定何时对哪些数据进行访问。例如，主站设备可以使用短循环周期刷新驱动器中数据，长循环周期采样 I/O 端口，固定的过程数据结构不是必要的。这也使得 EtherCAT 主站设备相较于传统的现场总线系统减轻了负担，在传统的现场总线系统中，主站需要单独读取每个节点

Ethercat 应用场景

1.运动控制器/运动控制卡

脉冲型升级为 EtherCAT 总线系统，提高性能和降低成本。

应用在多轴控制系统中，大幅度降低成本和简约电气布线。

分布式从站节点，搭配自由方便灵活。

2.数控系统 CNC

脉冲型升级为 EtherCAT 总线系统，增加系统稳定性、提高性能和降低成本。

摆脱由于传统脉冲频率的限制，体现高精高速的性能需求。

3.机器人/机械手臂等多轴高精度控制系统

降低 CPU 使用率，让 CPU 更加出色做高阶算法。

电气简单，控制稳定。

分布式从站节点，搭配自由方便灵活。

4.工控主板

支持目前主流现场工业以太网总线，提高性能，增加其附加值。

5.现场工业总线协议转换

EtherCAT 协议转换到其他协议总线方便快捷。

大型灯光舞台控制系统及 5D 影院控制系统

二、EtherCAT 从站

2.1 产品概述

使用紫光 FPGA 逻辑实现 EtherCAT 协议，实现从站功能。更加突出了 EtherCAT 现场总线的同步性能及高效性，大量减轻处理器的负担，CPU 只需要专注于计算，无需关注 ethercat 发包。只需要将数据通过 FSMC 协议写入 FPGA。FPGA 会把数据打包成 EtherCAT 格式。通过网络发送给下一个从站。

如图所示是 EtherCAT 从站（带 MCU 芯片）样卡图 2.1.1 是正面图，图 2.1.2 是反面图。

图 2.1.1

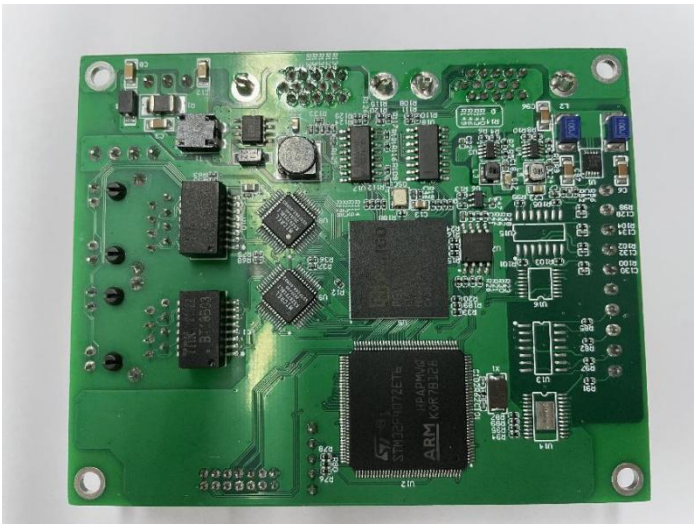


图 2.1.2

同时也做了另外一款样卡，在同创 FPGA 中实现了 IO 功能，如图所示是 EtherCAT 从站（IO 卡）样卡图 2.1.3 是正面图，图 2.1.4 是反面图。用户可以根据自己选择，使用不同封装 FPGA，实现更多引脚做 IO 功能。

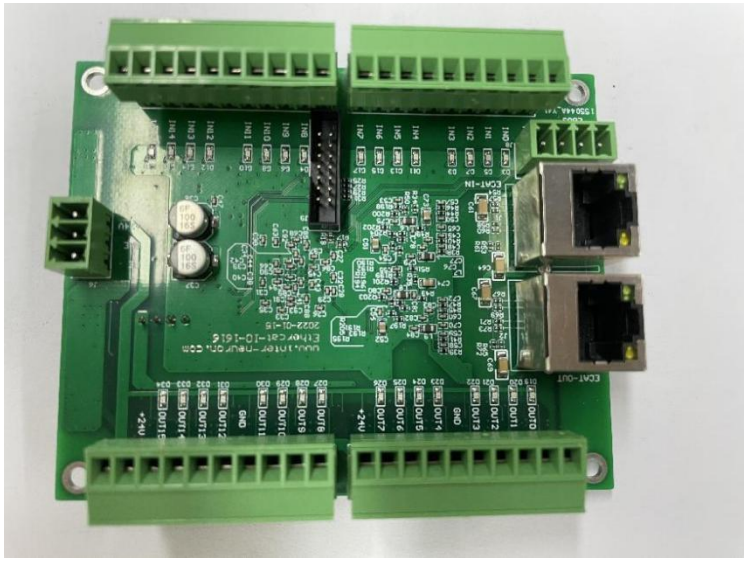


图 2.1.3

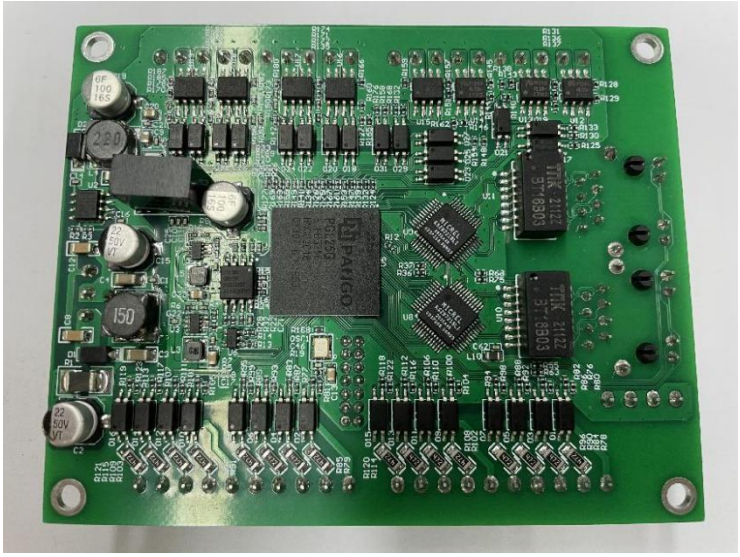


图 2.1.4

EtherCAT 从站实现了标准的 EtherCAT 协议，采用通用的 Memory 接口，可以连接任何 CPU 对 IP 进行控制。可用于任何标准的 EtherCAT 电机、IO 控制完全按照 EtherCAT 协议规范进行开发，整个报文的组装和解析完全在 FPGA 内进行。

对于 cpu 来说，完全可以把 FPGA 看成一个 SRAM 设备。Cpu 和 IP 进行数据的通信，会把这些报文按照一定的要求组装成 EtherCAT 报文发送出去。这样可以大大简化 cpu 侧软件的开发。数据同步以及重试、线缆冗余等功能也直接在 FPGA 内完成。这样可以减轻 CPU 侧的负担，CPU 只需专注于上层软件的开发，无需再关心底层的通信。

产品特性：

1) 集成数据帧转发处理单元，通信性能不受从站微处理器性能限制。最多可提供 4 个数据收发端口；

2) 最大 64K 字节双端口存储器 DPRAM 存储空间，其中包括 4K 字节的寄存器空间和 1~60K 字节的用户数据区，DPRAM 可由外部微处理器使用并行或串行数据总线访问，访问 DPRAM 称物理设备接口 PDI(Physical Device Interface)。

3) 可不用微处理器控制，作为数字量输入/输出芯片独立运行，具有通信状态机处理功能，最多提供 32 位数字量输入输出。

4) 具有 FMMU 逻辑地址映射功能，提高数据帧利用率。

5) 由储存同步管理器通道 SyncManager(SM)管理 DPRAM，保证应用数据的一致性和安全性。

6) 集成分布时钟(Distribute Clock)功能，为微处理器提供高精度中断信号。

7) 具有 EEPROM 访问功能，存储和应用配置参数，定义从站信息接口(SII, Slave Information Interface)。

相比 ASIC 优点：

1. EtherCAT 从站 IP 仅占用资源 15k 左右，FPGA 芯片剩余资源可作它用

2.相比传统 ASIC 方案更具价格优势

3.相比传统 ASIC 从站，IO 扩展更加灵活，选择不同封装 FPGA 可获得更多 IO 控制。

2. 4.硬件布局布线上更加灵活，当传统方案需要 ASIC+FPGA，该方案只需一块 FPGA 芯片，节省板卡空间和布局布线。

如图所示，EtherCAT 从站和其它 ASIC 对比

产品型号 特性	ET1100	LAN9252	本案（ZD7800）
端口	2~4（每个端口EBUS/MII）	2个内部PHY 1个MII	2~4（每个端口EBUS/MII）
FMMU	8	3	3~8可选
同步管理器	8	4	4~8可选
RAM(KB)	8	4	8K
分布时钟	64位	64位	64位
过程数据接口	同步异步16bit/同步异步8bit/spi/32位数字IO	异步16bit/异步8bit/spi/16位数字IO	异步16bit/异步8bit/spi/32位数字IO

从站支持标准 EtherCAT 两种物理层接口模式：MII 和 EBUS。MII 是标准以

太网物理层接口，使用外部物理层芯片，一个端口传输延时约为 500ns。EBUS 是使用 LVDS 标准定义的数据传输标准，可直接连接 ESC 芯片，不需要额外物理层芯片，从而避免物理层附加传输延时，一个端口传输延时约为 100ns。EBUS 最大传输距离只有 10m（建议不超过 0.5m），适用于距离较近 I／O 设备或伺服驱动器之间的连接。

2.2 EtherCAT 从站整体框架

采用紫光 FPGA 和 MCU 配合完成 EtherCAT 从站功能，如图 2.2.1 所示。MCU 通过 FSMC 协议和 FPGA 进行交互，可以把 FPGA 看成是一个 SRAM 设备。MCU 读写内存等时，会把内存相应数据通过 FSMC 协议发给 FPGA。

FPGA 拿到数据后对数据进行 EtherCAT 协议组包、解包，并把数据通过网口发给下一个 EtherCAT 从站。和传统从站相比 FPGA 充当了 ASIC，MCU 只需要关注业务运算。无需关心 EtherCAT 协议组包和解析。

由 MCU 设置 FPGA 参数，让 FPGA 周期性出发中断告知 MCU 进行业务处理并且把数据通过 FSMC 协议和 FPGA 进行交互。

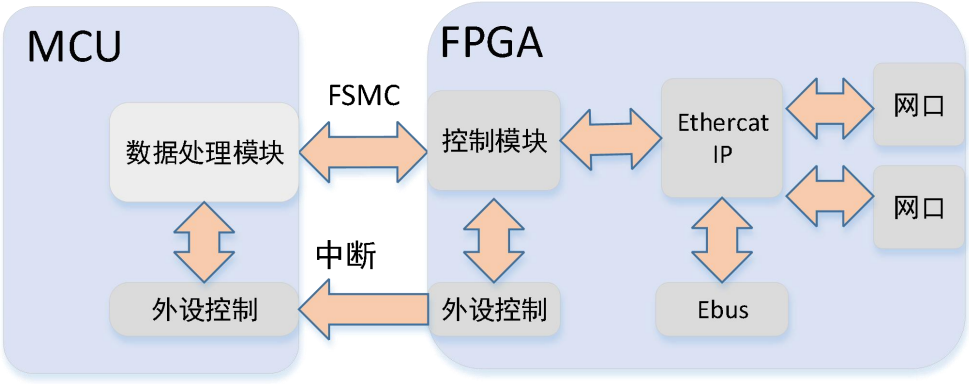


图 2.2.1

EtherCAT 从站接入网络连线图如图 2.2.2 所示，一个主站可以控制多个从站。只需要把运算好数据根据接口函数写入 FPGA 中。FPGA 会把数据通过网口同步给

各个从站，同时也会把从站反馈的数据进行解包。存放在 FPGA 内部 RAM 当中。软件只需通过给到函数接口读取即可。

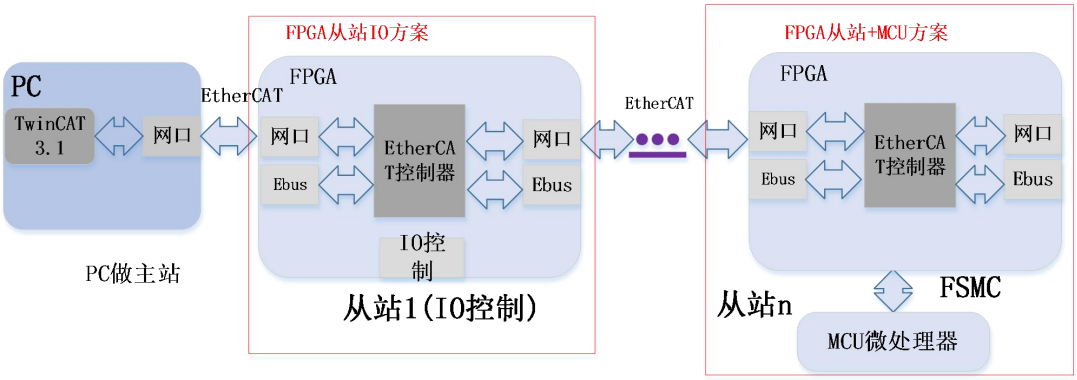


图 2.2.2

2.3 EtherCAT 套件交互

硬件：

1. 从站开发板 PGL25G（27K LUT4）一块
2. 从站参考原理图及 PCB 2 份

软件：

1. MCU 参考代码 2 份
2. EtherCAT Slvae IP 网表文件
3. FPGA 参考工程

文档：

1. EtherCAT 从站软件开发指南一份

2.4 搭建二次开发环境

1、硬件接线环境搭建

前提条件

24V 电源、 网线、 EtherCAT 从站、 win7 系统以上电脑一台

注意事项

网口有两个，一进一出，进是连接上游，出是连接下游。切勿连错

操作步骤

如下图 2.3.1 所示：

1. 首先将 win7 电脑网线口引出，接入 EtherCAT 从站卡入口，如果没有下一个从站,EtherCAT 从站出口无需接。

2.图中 24V 电源接口处，接入 24V 电源。EtherCAT 从站根据对应电源链路接入

3.上电即可完成整个运行环境搭建

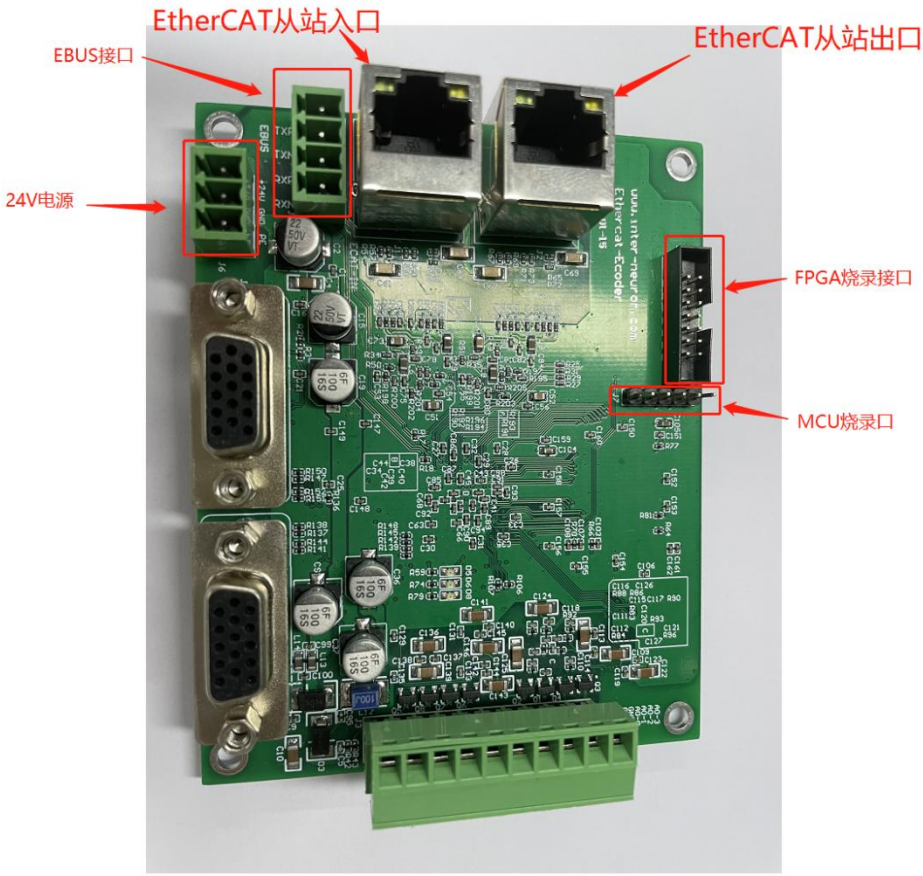


图 2.4.1

2、开发环境搭建

EtherCAT 从站出厂自带 MCU 和 FPGA 程序。Xml 文件已经烧录进 eeprom，通过 TwinCAT 可以直接扫描从站。

TwinCAT 环境搭建

根据如下连接下载 TwinCAT 安装包进行安装，安装完毕后根据后续 TwinCAT 操作就可以识别从站卡。TwinCAT 安装包如下：

链接：https://pan.baidu.com/s/1leT_Rr4SBUqOmFrjVIOYMw?pwd=1234
提取码：1234

MCU 软件运行环境搭建

下面是百度云链接，安装 keil 软件及 MCU 补丁包。安装完成后才可以打开 MCU 工程。可以在提供参考工程上进行业务开发。

链接：https://pan.baidu.com/s/1EXdQivAb3OSTZp6WEc04SQ?pwd=1234
提取码：1234

FPGA 开发软件安装

由于从站使用的是紫光同创 FPGA，当用户需要对 FPGA 进行业务开发时，涉及使用紫光同创 FPGA 软件及 license。请联系人员。申请 FPGA 开发软件的 license 及软件使用支持。

出厂板卡 FPGA 中内置 EtherCAT 从站程序，如果用户不涉及 FPGA 或者不需要修改 FPGA 程序，请忽略此项。

MCU 程序目录架构

EtherCAT 从站 MCU 程序一共有两个，wm_lan9252_CiA402_20171001、wm_lan9252_IO_20171109 当拿到 EtherCAT 从站程序后，进行解压 MCU 控制程序后得到如下目录。

MCU 硬件相关目录： Libraries User

工程目录： Project

业务开发目录： EtherCat

用户开发目录代码主要是业务开发目录里面的代码，在里面根据所需业务功能进行修改即可

FPGA 程序目录框架（如不涉及 FPGA 使用，请忽略此项）

EtherCAT 从站的 FPGA 工程一共有两个，一个是 EC_IO_1010_FPGA、一个是 EC_Slave_25G。用户拿到 FPGA 工程后，FPGA 代码文件主要如下所示。当需要对 FPGA 进行开发，可以参考我们样例，对 FPGA 程序进行增加，我们参考工程调用网表部分无需要变动。

EtherCAT IO 卡（纯 FPGA）主要代码如下：

从站网表：EthercatSlavePro.vm

从站应用程序：EC_MasterFpga.v

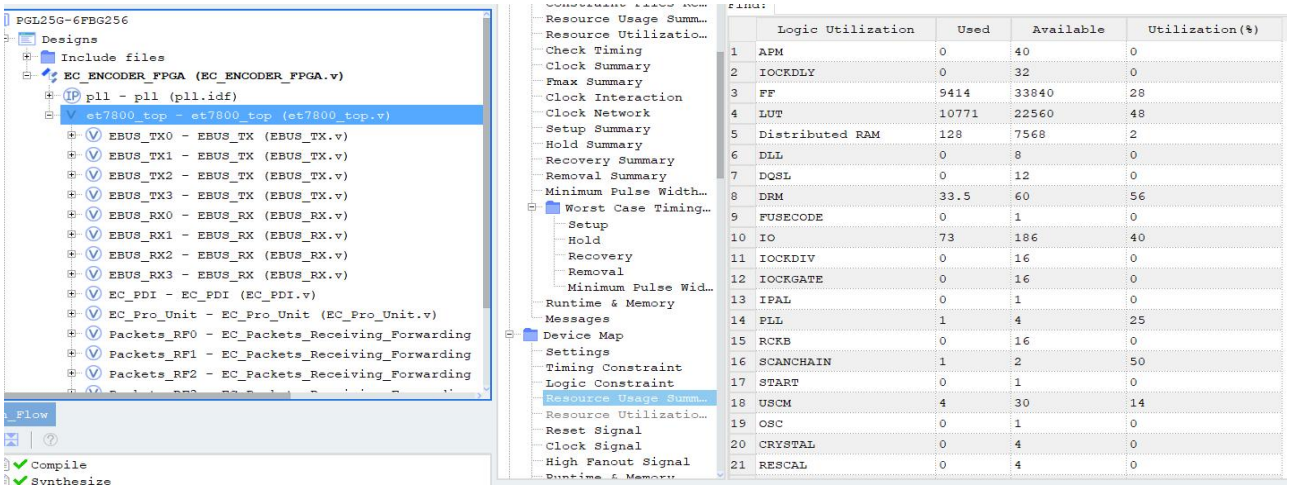
EtherCAT MCU+FPGA 卡主要代码如下：

从站网表：EthercatSlavePro.vm

从站应用程序:EC_ENCODER_FPGA.v

2.5 从站在 FPGA 资源占用情况

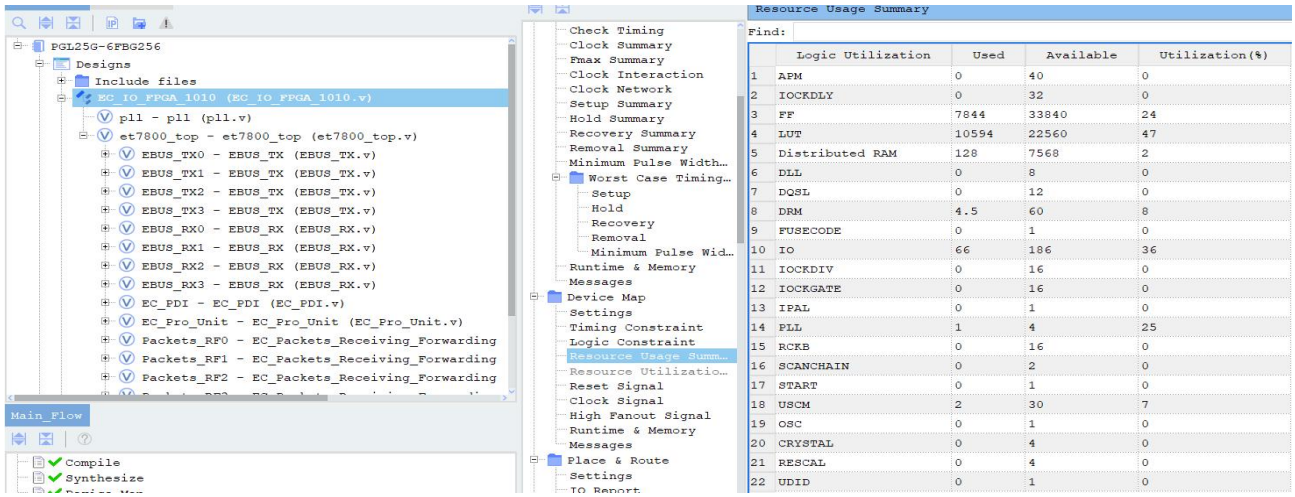
纯 IO 卡 FPGA 实现 EtherCAT 从站功能资源占用率如下图 2.6.1 ，在紫光同创 PGL-25G 上实现。占用资源为 10K LUT5，总共资源是 22K LUT5.占用资源大概为 47%



	Logic Utilization	Used	Available	Utilization(%)
1	APM	0	40	0
2	IOCKDLY	0	32	0
3	FF	9414	33840	28
4	LUT	10771	22560	48
5	Distributed RAM	128	7568	2
6	DLL	0	8	0
7	DQSL	0	12	0
8	DRM	33.5	60	56
9	FUSECODE	0	1	0
10	IO	73	186	40
11	IOCKDIV	0	16	0
12	IOCKGATE	0	16	0
13	IPAL	0	1	0
14	PLL	1	4	25
15	RCKB	0	16	0
16	SCANCHAIN	1	2	50
17	START	0	1	0
18	USCM	4	30	14
19	OSC	0	1	0
20	CRYSTAL	0	4	0
21	RESCAL	0	4	0

图 2.6.1

在 MCU+ FPGA 样卡上实现 EtherCAT 从站功能，资源占用率如下图，在紫光同创 PGL-25G 上实现。也是占用资源为 10K LUT5，总共资源是 22K LUT5.占用资源大概为 46.9%

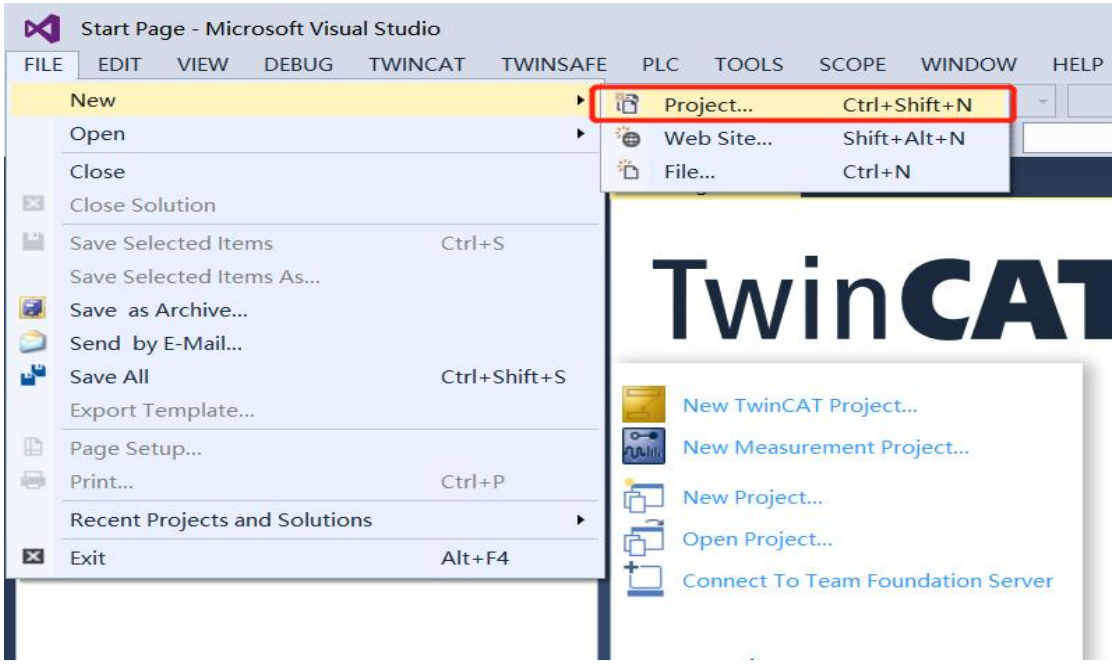


	Logic Utilization	Used	Available	Utilization(%)
1	APM	0	40	0
2	IOCKDLY	0	32	0
3	FF	7844	33840	24
4	LUT	10594	22560	47
5	Distributed RAM	128	7568	2
6	DLL	0	8	0
7	DQSL	0	12	0
8	DRM	4.5	60	8
9	FUSECODE	0	1	0
10	IO	66	186	36
11	IOCKDIV	0	16	0
12	IOCKGATE	0	16	0
13	IPAL	0	1	0
14	PLL	1	4	25
15	RCKB	0	16	0
16	SCANCHAIN	0	2	0
17	START	0	1	0
18	USCM	2	30	7
19	OSC	0	1	0
20	CRYSTAL	0	4	0
21	RESCAL	0	4	0
22	UDID	0	1	0

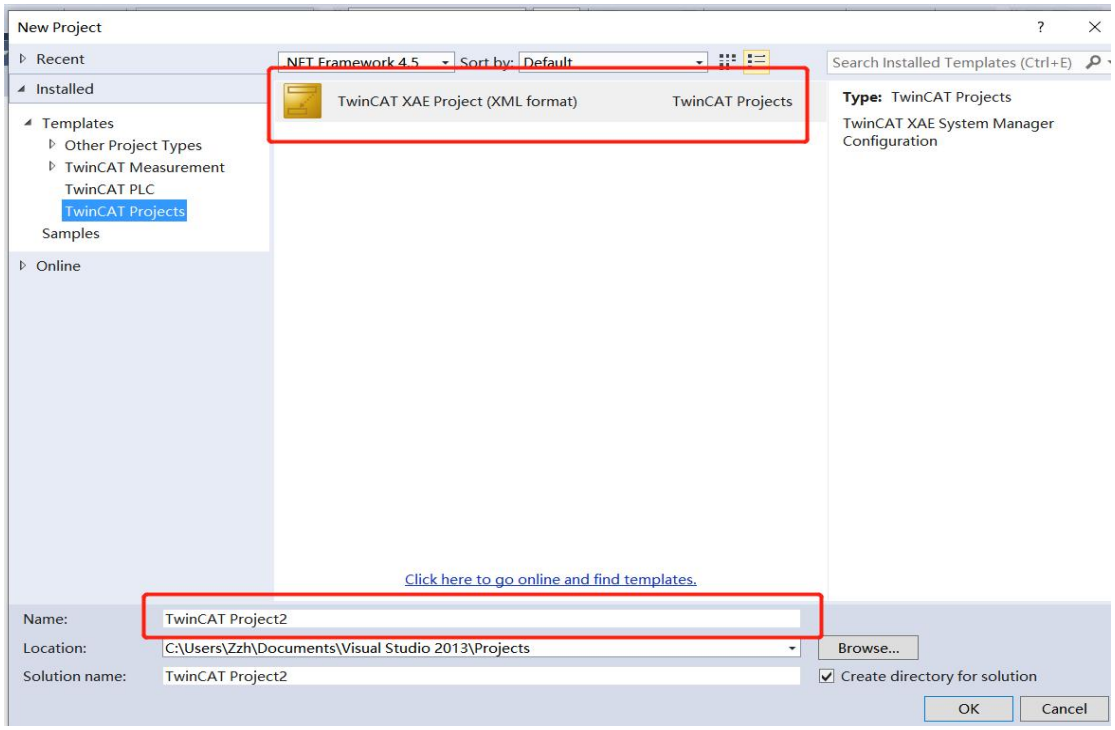
三、TwinCAT 相关操作

1、TwinCAT 扫描从站

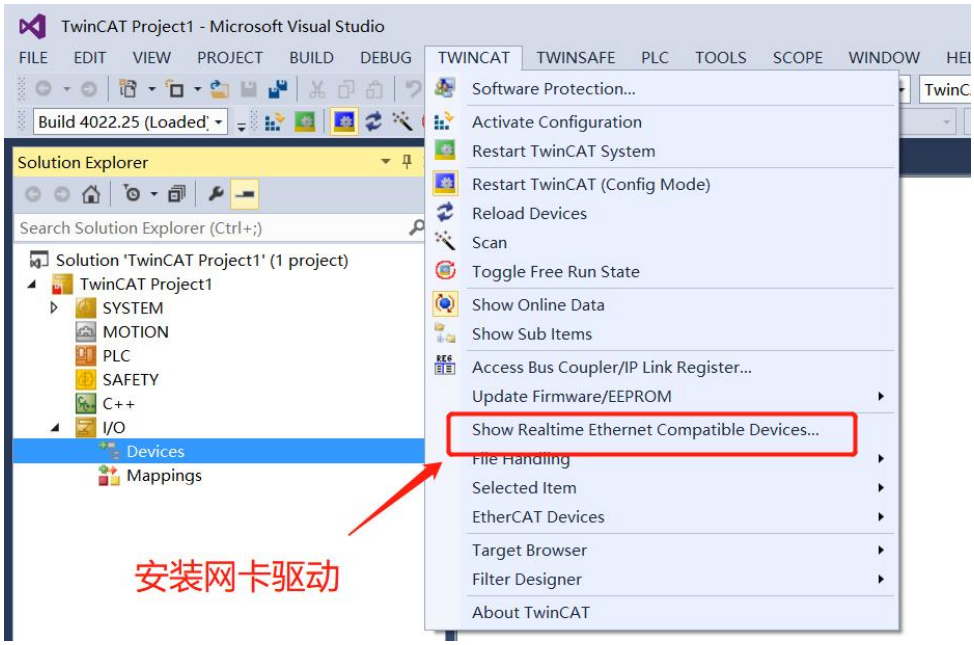
如图所示，创建工程



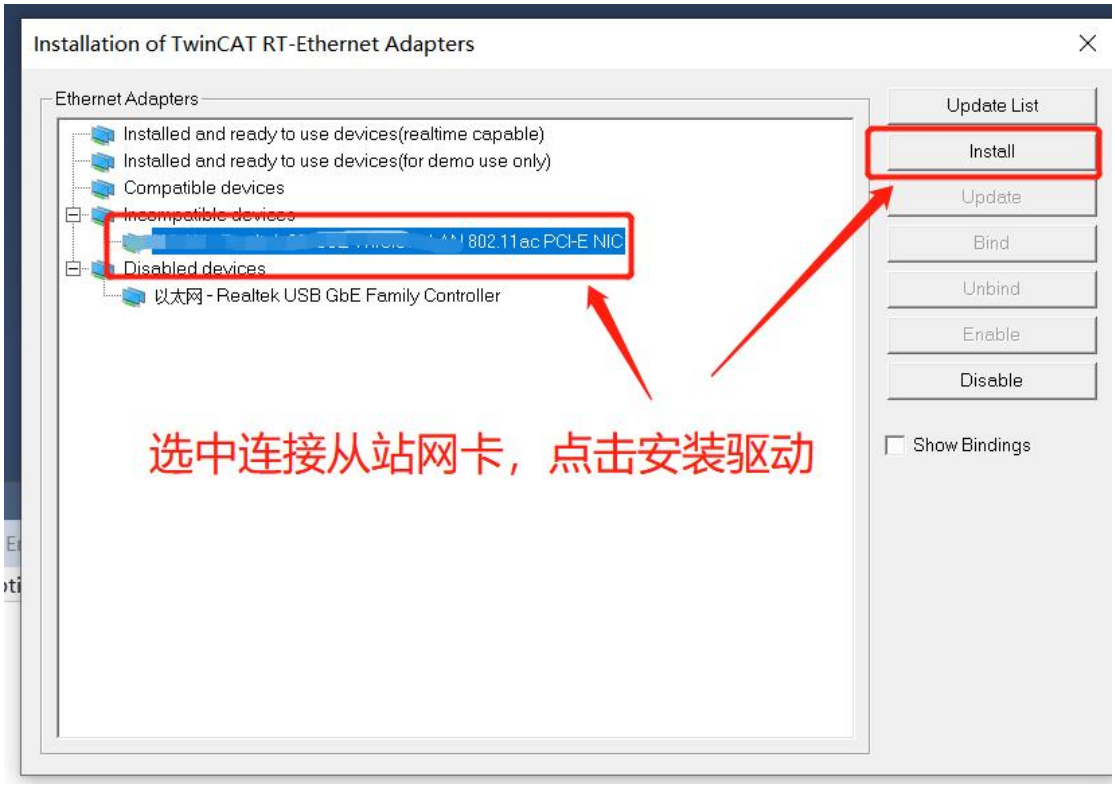
如图点击 TwinCAT 创建工程



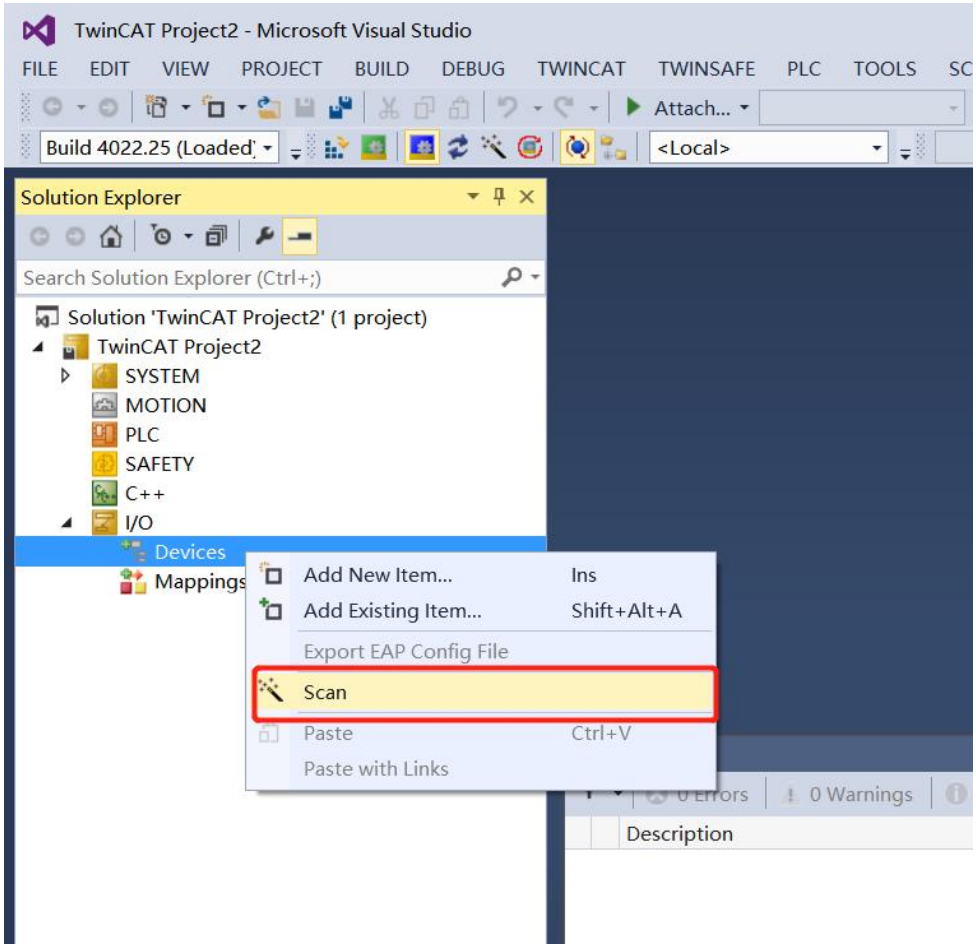
如图所示，点击安装网卡驱动



点击连接从站的 PC 网卡，选中安装驱动

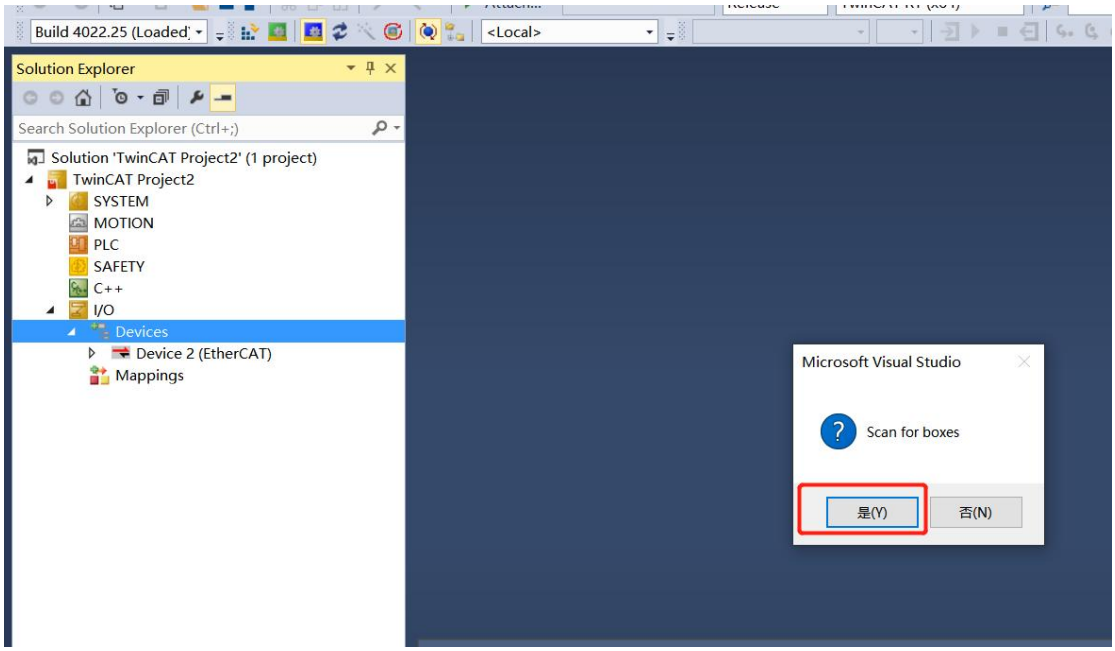


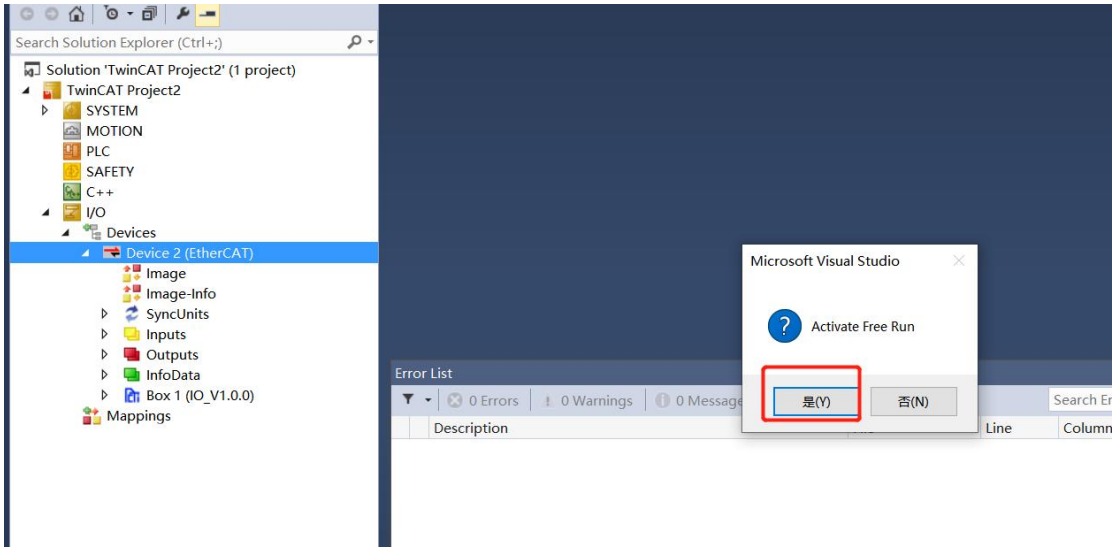
点击扫描从站


TwinCAT Project2 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TWINCAT TWINSAFE PLC TOOLS SC
Attach...
Build 4022.25 (Loaded)
<Local>
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'TwinCAT Project2' (1 project)
TwinCAT Project2
SYSTEM
MOTION
PLC
SAFETY
C++
I/O
Devices
Mappings
Add New Item... Ins
Add Existing Item... Shift+Alt+A
Export EAP Config File
Scan
Paste Ctrl+V
Paste with Links
0 Warnings
Description

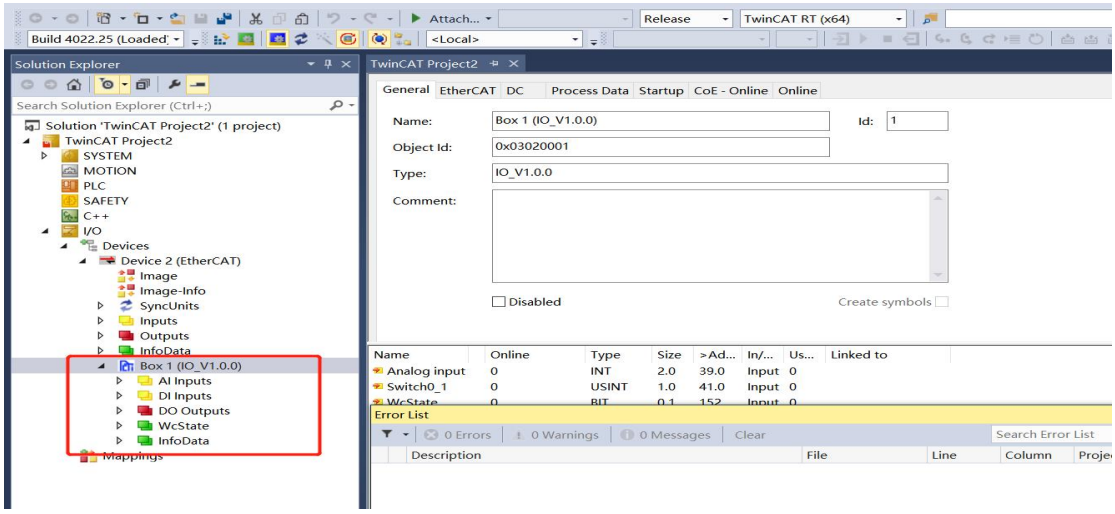
选中网卡，点击 OK


1 new I/O devices found
Device 2 (EtherCAT) [以太网 (Realtek USB GbE Family Controller)]
OK
Cancel
Select All
Unselect All
实际连接从站网卡


Build 4022.25 (Loaded)
<Local>
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'TwinCAT Project2' (1 project)
TwinCAT Project2
SYSTEM
MOTION
PLC
SAFETY
C++
I/O
Devices
Device 2 (EtherCAT)
Mappings
Microsoft Visual Studio
Scan for boxes
是(Y)
否(N)


Search Solution Explorer (Ctrl+;)
Solution 'TwinCAT Project2' (1 project)
TwinCAT Project2
SYSTEM
MOTION
PLC
SAFETY
C++
I/O
Devices
Device 2 (EtherCAT)
Image
Image-Info
SyncUnits
Inputs
Outputs
InfoData
Box 1 (IO_V1.0.0)
Mappings
Microsoft Visual Studio
Activate Free Run
是(Y)
否(N)
Error List
0 Errors
0 Warnings
Description
Line
Column

如图所示完成扫描


Attach...
Release
TwinCAT RT (x64)
Build 4022.25 (Loaded)
<Local>
Solution Explorer
TwinCAT Project2
General
EtherCAT
DC
Process Data
Startup
CoE - Online
Online
Name:
Box 1 (IO_V1.0.0)
Id:
1
Object Id:
0x03020001
Type:
IO_V1.0.0
Comment:
Disabled
Create symbols
Box 1 (IO_V1.0.0)
AI Inputs
DI Inputs
DO Outputs
WcState
InfoData
Name
Online
Type
Size
Analog input
Switch0_1
Error List
0 Errors
0 Warnings
0 Messages
Clear
Description
File
Line
Column

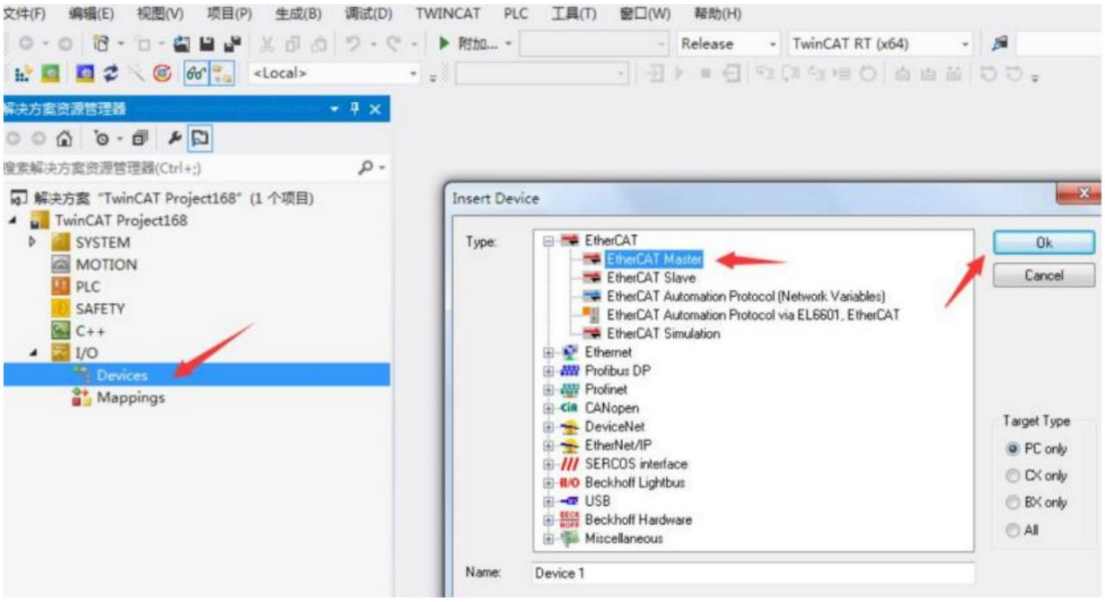
2、生成 XTI 操作步骤

1. 将所需要连接的从站的 xml 文件复制到 Twincat 软件的\3.1\Config\Io\EtherCAT 目录下。

2.打开 Twincat，电机文件->新建->项目

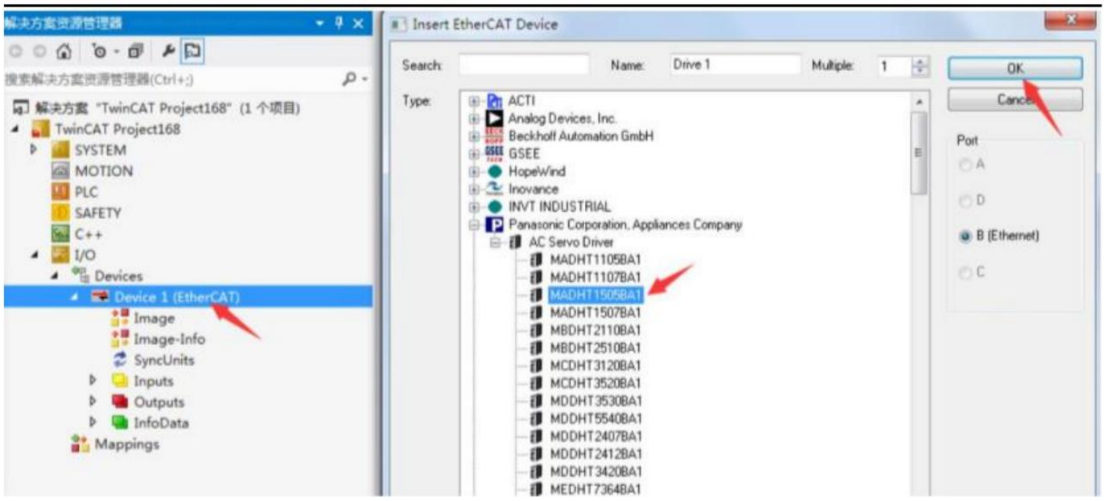
3.选择 TwincatXAEProject（XMLformat）,并点击确认

4.右键点击界面右侧的 Device,在弹出的菜单中点击添加新项，然后选择 EtherCATMaster 并点击 OK。在之后弹出的 Devicefoundat 菜单中，选择 none 即可



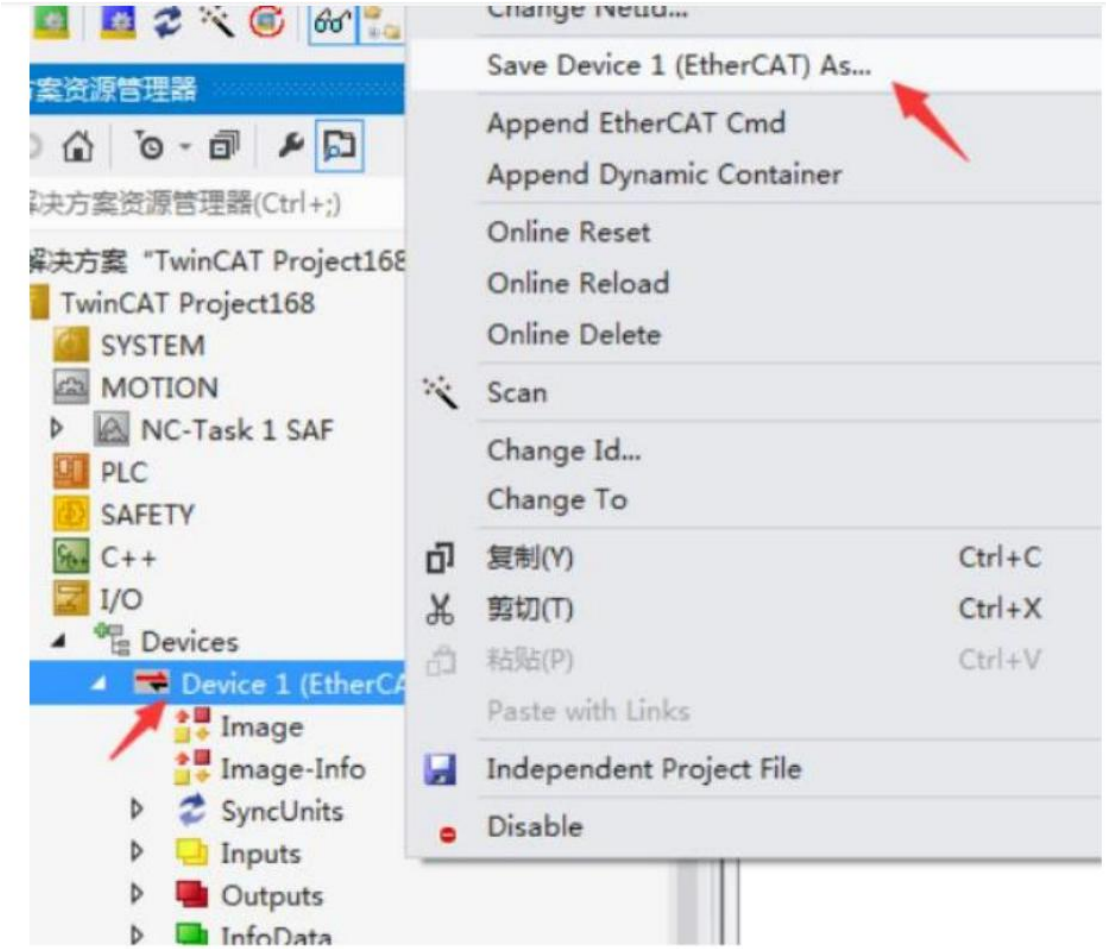
5.右键点击 Device1,并点击添加新项，在弹出的 InsertEtherCAT Device 选择

主站连接的第一个从站，之后点击 OK 添加。(本例中第一个从站为松下的 A5B 驱动器)



6.使用相同的方法依次添加第二个从站、第三个从站…最后一个从站。

7.右键点击 Device1(EtherCAT)，在弹出菜单栏中选择 SaveDevice1(EthercAT) As..，将配置保存成 xti 文件。

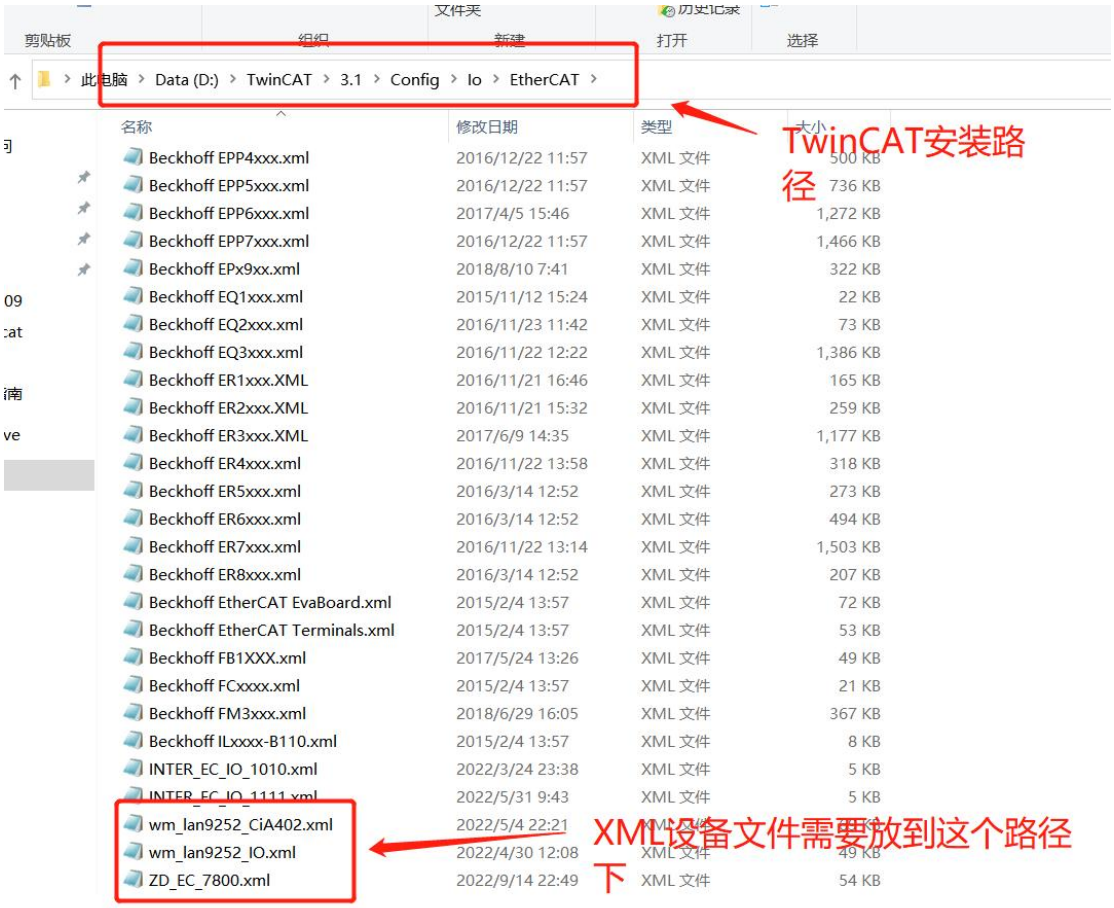


8.打开 XmlParser 并将保存的 xti 文件生成二进制文件。（生成的文件为 hexConf,手动将后缀修改为.bin）

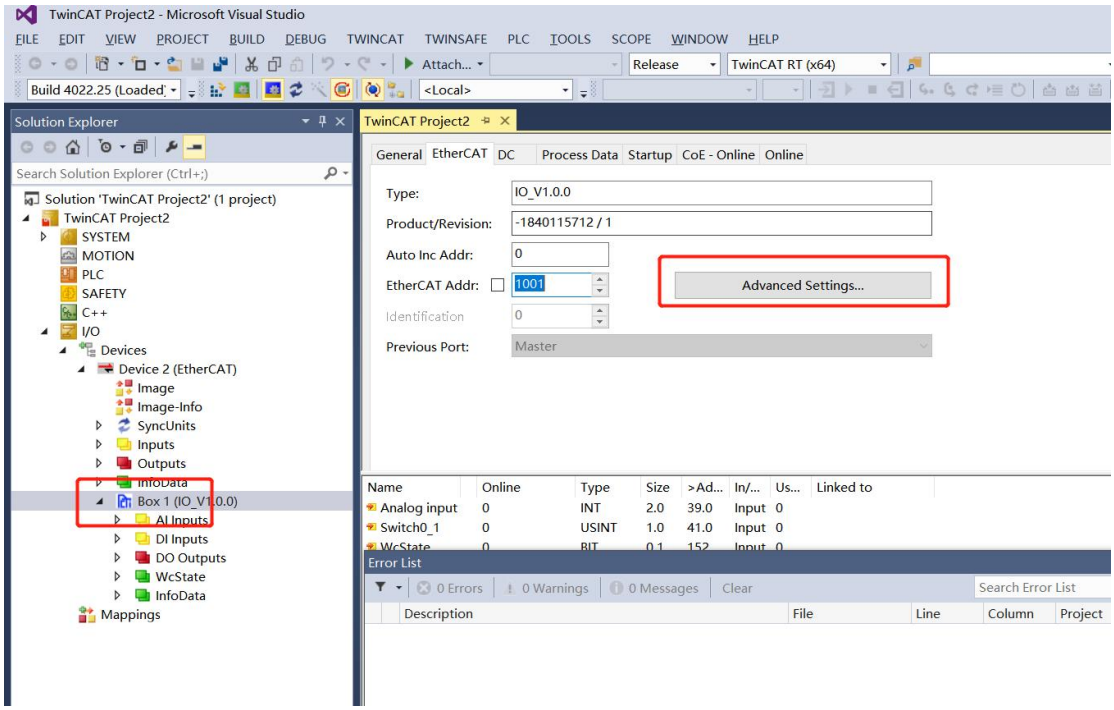
9.将.bin 二进制文件烧录到#defineXTI_FLASH_ADDR 0x08040000 定义的地址中。

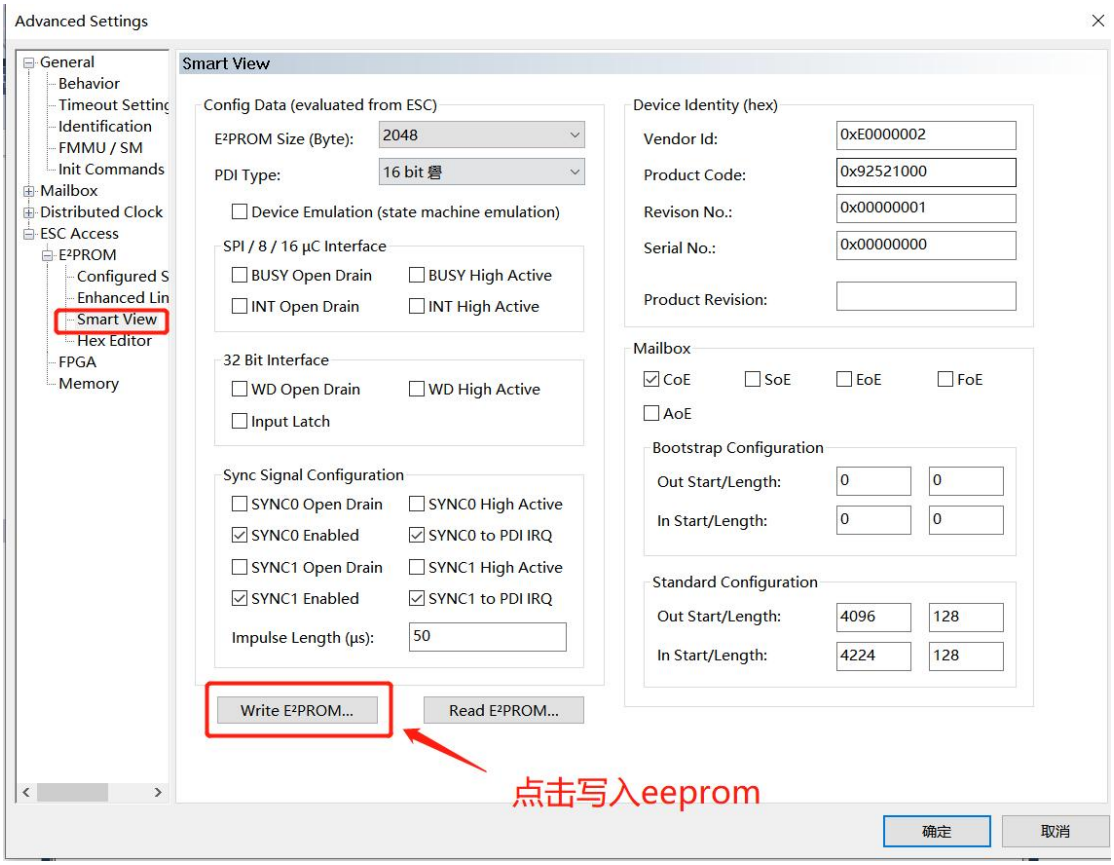
3、更新 xml 文件到从站 eeprom 操作

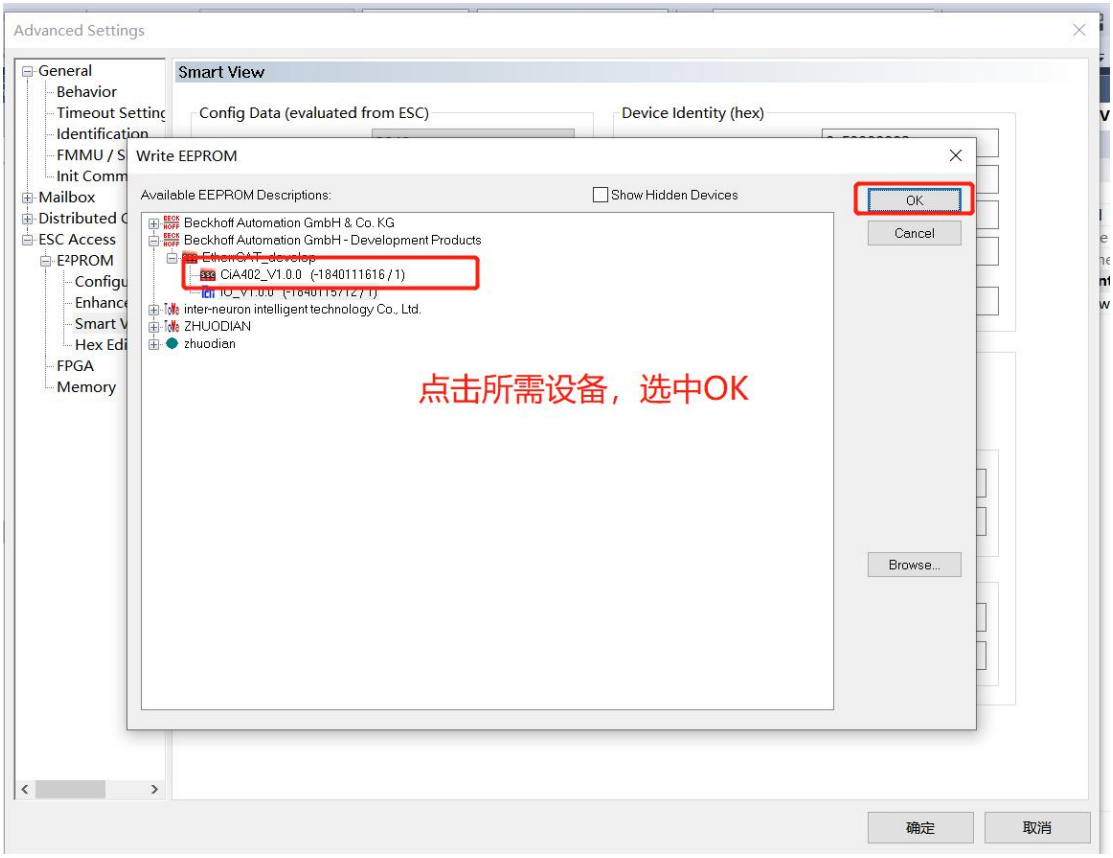
首先需要将我们提供得 XML 文件存放到如下路径



如图所示，通过上述**<TwinCAT 扫描从站>** ，把从站扫描上来。根据图片，把需要更新从站**-> EtherCAT -> advance settings**




Advanced Settings
General
Behavior
Timeout Setting
Identification
FMMU / SM
Init Commands
Mailbox
Distributed Clock
ESC Access
E²PROM
Configured S
Enhanced Lin
Smart View
Hex Editor
FPGA
Memory
Smart View
Config Data (evaluated from ESC)
E²PROM Size (Byte):
2048
PDI Type:
16 bit 譽
Device Emulation (state machine emulation)
SPI / 8 / 16 µC Interface
BUSY Open Drain
BUSY High Active
INT Open Drain
INT High Active
32 Bit Interface
WD Open Drain
WD High Active
Input Latch
Sync Signal Configuration
SYNC0 Open Drain
SYNC0 High Active
SYNC0 Enabled
SYNC0 to PDI IRQ
SYNC1 Open Drain
SYNC1 High Active
SYNC1 Enabled
SYNC1 to PDI IRQ
Impulse Length (µs):
50
Write E²PROM...
Read E²PROM...
Device Identity (hex)
Vendor Id:
0xE0000002
Product Code:
0x92521000
Revison No.:
0x00000001
Serial No.:
0x00000000
Product Revision:
Mailbox
CoE
SoE
EoE
FoE
AoE
Bootstrap Configuration
Out Start/Length:
0
0
In Start/Length:
0
0
Standard Configuration
Out Start/Length:
4096
128
In Start/Length:
4224
128
点击写入eeprom
确定
取消


Advanced Settings
Smart View
Config Data (evaluated from ESC)
Device Identity (hex)
Write EEPROM
Available EEPROM Descriptions:
Show Hidden Devices
OK
Cancel
Beckhoff Automation GmbH & Co. KG
Beckhoff Automation GmbH - Development Products
EthercAT_develop
CiA402_V1.0.0 (-1840111616 / 1)
IO_V1.0.0 (-1840115712 / 1)
inter-neuron intelligent technology Co., Ltd.
ZHUODIAN
zhuodian
点击所需设备，选中OK
Browse...
确定
取消

四、EtherCAT IP 使用指南

4.1 EtherCAT 模块概要

4.1.1 PHY 接线示意图

如图所示，EtherCAT IP，会输出 25MHz 时钟给到 PHY 芯片，PHY 芯片需要选用支持 MII 协议。PHY 完成自适应后，在 PHY_TX_CLK、PHY_RX_CLK 会给到相应时钟，需要注意 25MHz 才是正确时钟，如果是 2.5MHz 说明网口自适应可能失败。

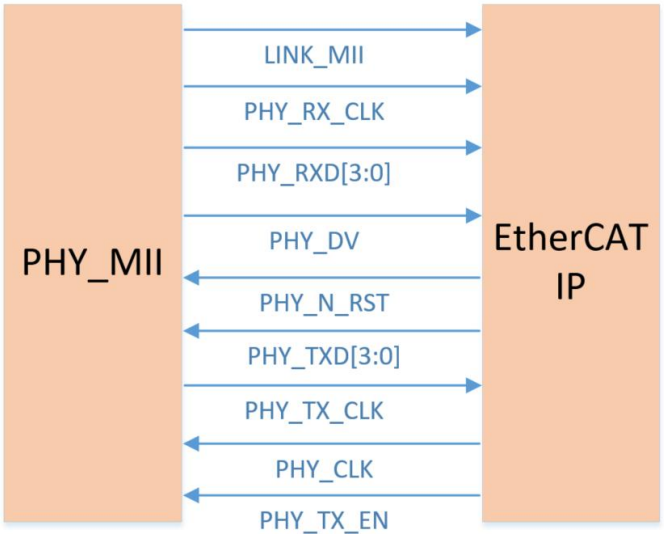


图 4.1.1.1

PHY 总体连接如图 4.1.2 所示，EtherCAT 从站需要两个 PHY，一进一出。PHY0 连接上一级从站 PHY1 或者连接主站网口，PHY1 连接下一级从站 PHY0 或者不连接（当为最后一个从站时）。PHY 的引脚接线根据我们推荐原理图接即可，在 FPGA 工程中关联硬件引脚，IP 会自动驱动 PHY 收发数据。

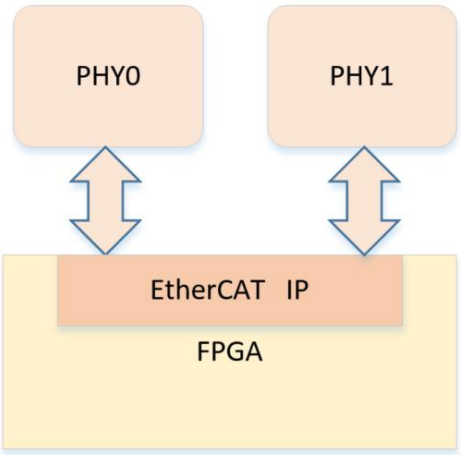


图 4.1.2

使用两个 PHY 就可以完成基本的 EtherCAT IP 通信，当用户需要更加高级功能，需要接 MCU 互联，让 MCU 进行一些相关运算。

4.1.2 MCU + FPGA 接线示意图

如图 4.1.2.1 所示，MCU 和 fpga 连接方式通过引脚进行互联。FSMC（Flexible static memory controller）接口用于 MCU 与 FPGA 之间的通讯。FPGA 模拟成 NAND Flash 作为 Memory 让 MCU 读写 FPGA 与 MCU 之间的连接方式如下所示：

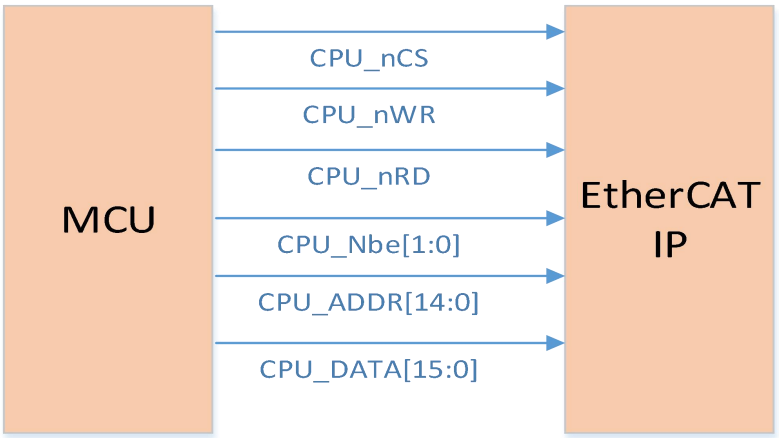


图 4.1.2.1

对于 FPGA 侧引脚定义如下表格所示

Pin 定义	类型	描述
CPU_DATA[15:0]	I/O	数据输入输出引脚 0~15

CPU_ADDR[14:0]	I	地址线 0~14
CPU_nCS	I	片选信号
CPU_nWR	I	写控制信号
CPU_nRD	I	读控制信号
CPU_nBE[1:0]	I	字节控制信号

4.1.3 EBUS/LVDS 物理层

EBUS 是一个 EtherCAT 物理层，用于降低组件和成本。它还减少了 ESC 内部的延迟。根据低压差信号（LVDS）接口电路的电气特性标准，EBUS 物理层使用低电压差信号（LVDS）。EBUS 有 100 Mbit/s 的数据速率来实现快速以太网数据速率。EBUS 协议只是封装了以太网帧，因此 EBUS 可以携带任何以太网帧。直接使用线缆或者通过 PCB 走线连接。

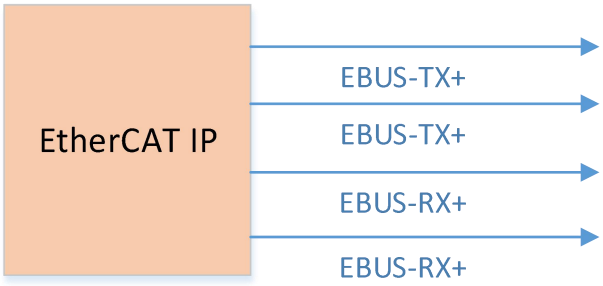


图 4.1.3.1

信号	描述
EBUS-TX+ EBUS-TX-	EBUS/LVDS 数据输出
EBUS-RX+ EBUS-RX-	EBUS/LVDS 数据输入

Ebus 只限于从站和从站短距离连接，如图 4.1.3.2

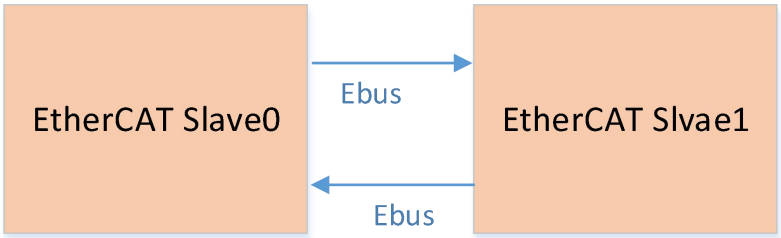


图 4.1.3.2

4.1.4 EEPROM

如图 4.1.4.1 所示，FPGA 中 ethercat IP 通过 i2c 和 eeprom 相连，IP 中默认 eeporm 设备地址为 0xA0.因此 eeprom 从设备地址 A0~A2 需要接地。在 IP 中可以指定 EEPROM 大小类型输入，默认 Eeprom 大小选用 0~16K.

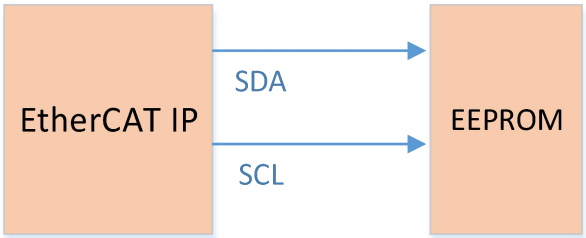


图 4.1.4.1

EtherCAT 从控制器存储设备配置和设备描述。EEPROM 与 I²C 接口和 FPGA 连接，控制速率为 100 多 K。上电完成后，FPGA 会通过 I2C 访问 eeprom。通过 eeprom 内容进行配置 EtherCAT IP，EEPROM 村粗配置如下表 4.1.4.2 所示

表 4.1.4.2

eeprom 地址	描述			
0x0	EtherCAT 从站控制器配置区域			
0x8	厂商 ID	产品 ID	RevisionNo	SerialNo
0x10	硬件延迟		引导邮箱配置	
0x18	邮箱同步管理配置		预留	
0x32	预留			
.....				

EEPROM 内容

ESC 配置区(EEPROM 字地址 0 到 7）在开机或复位后由 ESC 自动读取。它包含 PDI 配置、DC 设置和已配置的从站配置的一致性。EtherCAT 主服务器可以调用重新加载 EEPROM 内容。在这种情况下，它们只在开机或复位后的初始 EEPROM 加载时被接管。

具体前端详细介绍如表 4.1.4.3 所示,更多详细信息请参考如下链接：
http://www.ethercat.org

表 4.1.4.3

字段地址	描述	字段地址	描述
0x0	PDI 控制	0x14	引导接收邮箱偏移
0x1	PDI 配置	0x15	引导接收邮箱大小
0x2	同步信号的脉冲长度	0x16	引导发送邮箱偏移
0x3	扩展的 PDI 配置	0x17	引导发送邮箱大小
0x4	配置从站别名	0x18	标准接收邮箱偏移量
0x5~0x6	预留	0x19	标准接收邮箱偏大小
0x7	Checksum 校验	0x1A	标准发送邮箱偏移
0x8~0x9	Vendor ID	0x1B	标准发送邮箱大小
0xA~0xB	产品 ID	0x1C	邮箱协议
0xC:0xD	修订编号	0x1D:0x3D	预留
0xE:0xF	SN	0x3E	大小
0x10	执行延迟	0x3F	版本
0x11	端口 0 延迟	0x40	供应商特定类型
0x12	端口 1 延迟	0x41	以下类别 Word 大小
0x13	预留	0x42	类别数据

4.1.4 中断

EtherCAT 支持两种类型的中断：用于 MCU 的 AL 事件请求和专用于 EtherCAT 主站的 ECAT 事件请求。此外，分布式时钟同步信号也可以用作 MCU 的中断。

4.1.4.1 AL 事件请求(PDI 中断）

AL 事件请求可以使用 PDI 中断请求信号（IRQ/SPI_IRQ 等）向 MCU 发送信号。对于 IRQ 生成，AL 事件请求寄存器（0x0220:0x0223）与 AL 事件掩码寄存器（0x0204:0x0207）组合，

然后所有产生的位(逻辑 OR)组合成一个中断信号。IRQ 信号的输出驱动器特性可使用 SYNC/锁存器 PDI 配置寄存器（0x0151）进行配置。AL 事件掩码寄存器允许选择与 MCU 相关并由应用程序处理的中断。如图 4.1.4.1.1 所示

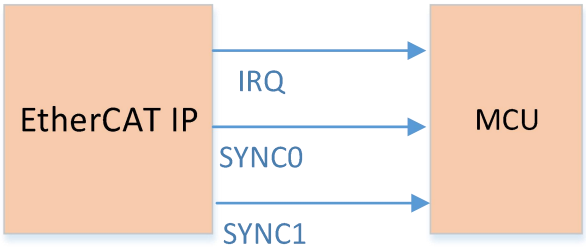


图 4.1.4.1.1

DC 同步信号可以通过两种方式用于中断生成：DC 同步信号被映射到 AL 事件请求寄存器（配置为 SYNC/锁存 PDI 配置寄存器 0x0151.3/7）。在这种情况下，**从 ESC 到 MCU 的所有中断都被合并成一个 IRQ 信号**，并且仍然可以使用分布式时钟 LATCH0/1 输入。IRQ 信号的抖动为 40 ns。DC 同步信号直接连接到 MCU 中断输入。

MCU 可以在 DC 同步信号中断时反应更快（不需要读取 AL 请求寄存器），但它需要更多的中断输入。同步信号的抖动为 12 ns。DC 锁存功能仅用于一个锁存输入（如果使用两个 DC 同步输出）。

用于 AL 事件请求的寄存器见表 4.1.4.2

表 4.1.4.1.2 注册为 AL 事件请求配置

寄存器地址	寄存器定义	描述
0x0150	PDI 配置	IRQ 驱动程序特性
0x0151	同步/锁存 PDI 配置	将 DC 同步信号映射到中断
0x0204:0x0207	AL 事件掩码	掩码寄存器
0x0220:0x0223	AL 事件请求	触发中断
0x0804 + N*8	同步管理器控制	映射同步管理器中断

4.1.4.2 ECAT 事件请求

ECAT 事件请求寄存器（0x0210：0x0211）与 ECAT 事件掩码寄存器（0x0200：0x0201）相结合。两者相与后得到触发中断编号。

ECAT 事件请求配置的注册器

寄存器地址	寄存器定义	描述
0x0200:0x0201	ECAT 事件掩码	掩码寄存器
0x0210:0x0211	ECAT 事件请求	中断寄存器

0x0804 + N*8	同步管理器控制	映射同步管理器中断

4.2 EtherCAT IP 接口描述

模块定义

```
module EthercatSlavePro(
                          Clk_100,
                         Ebus_Sclk,
                         Ebus_Sclk90,
                          Reset,
                        SoftReset,
                          Oclk25,
                          MiiTxShift,
                          LinkMii,
                          P_CONF,
                          //MII
                          TxClk,TxEn,
                          Txd0,Txd1,Txd2,Txd3,
                          RxClk,RxDv,RxEr,
                          Rxd0,Rxd1,Rxd2,Rxd3,
                          //EBUS
                          Ebus_RxLvds,
                          Ebus_TxLvds,
                          //EEPROM
                          EepromSize,
                          EepromClk,
                          EepromData_In,
                          EepromData_Out,
                          EepromData_Dir,
                          //STATUS
                          EepromLoaded,
                          CPU_IRQ,
                          WD_TIRG,
                          SOF,
                          //IO
                          DigitalOutPut,
                          DigitalInput,
                          LatchIn,
                          OutValid,
                          //SPI
```

```
                                    SPI_SEL,
                                    SPI_CLK,
                                    SPI_DI,
                                    SPI_DO,
                                    //ASYNC
                                    CPU_nCS,
                                    CPU_nWR,
                                    CPU_nRD,
                                    BHE,
                                    CPU_ADDR,
                                    CPU_IDATA,
                                    CPU_ODATA,
                                    CPU_nBUSY,
                                    iSyncLatch,
                                    oSyncLatch,
                                    SyncLatchDir,//1:out,0:in
                                    //LED
                                    RunLed
                                );
input                   Clk_100;
input                   Ebus_Sclk,Ebus_Sclk90;
input                   Reset;
output                  SoftReset;
output                  Oclk25;
input     [1:0]         MiiTxShift;
input     [3:0]         LinkMii; //PHY signal indicating a link,1:be good for link; 0:Disconnect
input     [3:0]         P_CONF;
input     [3:0]       TxClk;
output    [3:0]       TxEn;
output    [3:0]       Txd0,Txd1,Txd2,Txd3;
input     [3:0]       RxClk;
input     [3:0]       RxDv;
input     [3:0]       RxEr;
input     [3:0]       Rxd0,Rxd1,Rxd2,Rxd3;
input     [3:0]       Ebus_RxLvds;
output    [3:0]       Ebus_TxLvds;
input                   EepromSize;
output                  EepromClk;
input                   EepromData_In;
output                  EepromData_Out;
output                  EepromData_Dir;
output                  RunLed;
output                  EepromLoaded;
output                  CPU_IRQ;
```

```
//digitalIO
output    [31:0]   DigitalOutPut;
input     [31:0]   DigitalInput;
input                  LatchIn;
output                   OutValid;
output                   SOF;
output                   WD_TIRG;
//SPI
input                  SPI_SEL;
input                    SPI_CLK;
input                    SPI_DI;
output                   SPI_DO;
//ASYNC MCU
input                  CPU_nCS;
input                  CPU_nWR;
input                    CPU_nRD;
input              BHE;
input    [15:0]    CPU_ADDR;
input    [15:0]    CPU_IDATA;
output [15:0]      CPU_ODATA;
output                   CPU_nBUSY;
input     [1:0]    iSyncLatch;
output    [1:0]    oSyncLatch;
output    [1:0]    SyncLatchDir;
endmodule
```

IP 端口说明

信号定义	定义描述
Clk_100	100M 时钟输入
Clk_Ebus	25M 时钟输入
nRst	复位信号输入，低电平有效
Oclk25	25M 时钟输出，用于提供 PHY 时钟使用
MiiTxShift[1:0]	MII 发送信号移相设置输入，0 正常对齐，1 信号延时 10ns，2 信号延时 20ns，3 信号延时 30ns
LinkMii[3:0]	PHY0~3 端口连接信号输入，高有效（仅当选择 MII 接口时使用，否则空即可）
P_CONF[3:0]	网络接口 0~3 类型选择输入，1 选择 MII 接口，0 选择 EBUS 接口
TxClk[3:0]	MII 接口 0~3 的发送时钟输入（仅当选择 MII 接口时使用，否则直接输入 0）
TxEn[3:0]	MII 接口 0~3 的发送使能输入（仅当选择 MII 接口时使用，否则空即可）

Txd0[3:0],Txd1[3:0], Txd2[3:0],Txd3[3:0]	分别是 MII 接口 0~3 的发送数据输出仅当选择 MII 接口时使用，否则空即可）
RxClk[3:0]	MII 接口 0~3 的接收时钟输入（仅当选择 MII 接口时使用，否则直接输入 0）
RxDv[3:0]	MII 接口 0~3 的接收有效信号输入（仅当选择 MII 接口时使用，否则直接输入 0）
RxEr[3:0]	MII 接口 0~3 的接收错误信号输入（仅当选择 MII 接口时使用，否则直接输入 0，当选择 MII 接口硬件没有连接也需要直接输入 0）
Rxd0[3:0],Rxd1[3:0], Rxd2[3:0],Rxd3[3:0]	MII 接口 0~3 的接收数据输入（仅当选择 MII 接口时使用，否则直接输入 0）
Ebus_RxLvds[3:0]	EBUS 接口 0~3 接收输入信号（仅当选择 EBUS 接口时使用，否则直接输入 0）
Ebus_TxLvds[3:0]	EBUS 接口 0~3 接收发送信号（仅当选择 EBUS 接口时使用，否则空即可）
EepromSize	EEPROM 大小类型输入，0 为 1~16kbits，1 为 32k~4M
EepromClk	EEPROM 时钟输出
EepromData_In	EEPROM 数据输入
EepromData_Out	EEPROM 数据输出
EepromData_Dir	EEPROM 数据方向输出，0 输入，1 输出。 注：请在顶层使用 EepromData 作为 EEPROM 的数据输入输出，类似下面的应用； EepromData = EepromData_Dir ? EepromData_Out : 1'bZ; EepromData_In = EepromData_Dir ? 1'b1 : EepromData;
EepromLoaded	EEPROM 已加载信号指示输出（不使用空即可）
CPU_IRQ	事件中断输出，高电平有效中断（不使用空即可）
DigitalOutPut[31:0]	数字 IO 输出（不使用空即可）
DigitalInput[31:0]	数字 IO 输入（不使用输入 0）
LatchIn	数字 IO 输入锁存输入信号（上升沿锁存到输入）
OutValid	数字 IO 输出有效信号输出，高电平有效
SOF	理器接收以太网帧开始标志输出，高脉冲有效
WD_TIRG	看门狗触发信号输出
SPI_SEL	SPI 片选输入，低电平有效
SPI_CLK	SPI 时钟输入
SPI_DI	SPI 数据输入

SPI_DO	SPI 数据输出
CPU_nCS	处理器异步片选输入，低电平有效
CPU_nWR	处理器异步读使能输入，低电平有效
BHE	处理器异步 16 位模式下的高位字节使能输入，低电平有效
CPU_ADDR[15:0]	处理器异步地址输入
CPU_IDATA [15:0]	处理器异步数据输入（使用异步 8 位处理器时使用低 8 位即可，高 8 位输入 0）
CPU_ODATA [15:0]	处理器异步数据输出（使用异步 8 位处理器时使用低 8 位即可，高 8 位空即可） 请在顶层使用 CPU_DATA 作为处理器数据输入输出，类似如下使用： CPU_DATA = (!CPU_nCS && !CPU_nRD)? CPU_ODATA : 16'hzzzz; CPU_IDATA = CPU_DATA;
CPU_nBUSY	处理器异步访问忙标志，低电平有效
iSyncLatch [1:0]	锁存器信号输入，与 oSyncLatch 共用
oSyncLatch [1:0]	分布式时钟同步信号输出，与 iSyncLatch 共用

SyncLatchDir [1:0]	分布式时钟同步信号输出和锁存器信号输入选择信号输入，对应位为 1 时分布式时钟同步信号输出，对应位为 0 时锁存器输入信号输入。
	注：请在顶层使用 SyncLatch[1:0]作为分布式时钟同步信号输出/锁存器信号输入双向 IO，类似下面使用：
	SyncLatch[0] = SyncLatchDir[0] ? oSyncLatch[0] : 1'bz;
	SyncLatch[1] = SyncLatchDir[1] ? oSyncLatch[1] : 1'bz;
	iSyncLatch[0] = ! SyncLatchDir[0] ? SyncLatch[0] : 1'b0;
	iSyncLatch[1] = ! SyncLatchDir[1] ? SyncLatch[1] : 1'b0;

RunLed	运行指示灯输出（不使用空即可）

五、常见问题解答

1、ethercat 从站同步抖动时长为多久？

答：从站抖动正负 80ns

2、ethercat 从站是否支持热拔插？

答：支持热插拔，需要重新初始化并建立连接。

3、ethercat 从站 EBUS 功能是否支持？

答：可以正常使用

4、需要经过 ETG 协会一致性认证。

答：一致性及稳定性是从站设备的需求，不是芯片通信协议的要求，这个需要做好从站设备自己找 etg 做认证。

5、一次性连接 24 个 ethercat 从站同步性能？最大延迟是多少

答：主要看 mcu 处理性能，如果只考虑通信，125us 完全可以实现。

6、从站资源是否还有优化空间？

答：从站支持可调功能，可根据实际需求，将资源控制在 15K-20K 范围。

7、ethercat 从站最低在紫光哪款 FPGA 使用

答：最低需要 PGL25G 以上

8、紫光 ethercat 从站和 asic 从价格上有区别吗？

答：传统 asic 从站价格在 70~120 ， 我们在 PGL25G 可以使用，价格优势显著

9、调试中碰到 PHY 和 FPGA 之间得 PHY_TX_CLK 和 PHY_RX_CLK 为 2.5M。

答:这样原因可能是 PHY 互联网口没插好。当 PHY 之间完成适配，是 25MHz 时钟。也是 EtherCAT IP 能正常工作的时钟。