# 帧格式序列检测模块规范说明书

模块名称：帧格式序列检测模块

版本信息：V1.0

编制日期：2025年04月27日

编制人：孟思文，张宇航，陈天宇

目录

[帧格式序列检测模块规范说明书 1](#_Toc201345967)

[1.引言： 3](#_Toc201345968)

[2.模块描述： 4](#_Toc201345969)

[2.1功能描述 4](#_Toc201345970)

[2.2测试要求 4](#_Toc201345971)

[2.3 关键性能指标 5](#_Toc201345972)

[3.接口定义： 6](#_Toc201345973)

[3.1顶层接口图 6](#_Toc201345974)

[3.2端口信号列表 6](#_Toc201345975)

[4.设计要求： 7](#_Toc201345976)

[4.1解帧模块： 7](#_Toc201345977)

[4.2 CRC计算模块： 11](#_Toc201345978)

[4.3 FIFO模块： 15](#_Toc201345979)

[4.4串行输出模块： 18](#_Toc201345980)

## 1.引言：

本规范文件详细描述了帧格式序列检测生成模块（Frame Format Sequence Detection and Generation Module）的设计目标、功能特性、性能指标以及测试要求。

该模块作为系统中通信协议栈的重要组成部分，负责在接收到的数据流中实时检测特定格式的帧序列，完成数据解帧、校验、缓存及编码输出等一系列处理，确保数据在多通道并行系统中的准确同步与分发。模块设计强调高性能、低延迟、协议灵活性，可广泛应用于高速数据通信、芯片内部数据通道控制、车载通信（如CAN、CAN-FD、CAN-XL协议）以及低延迟以太网（如10G Ethernet MAC）等应用场景。

在现代片上系统（SoC）设计中，随着数据通信速率不断提高，协议复杂度加大，传统串行数据通道已无法满足对实时性和可靠性的双重要求。因此，基于高性能帧检测与分发模块（如X\_CAN模块中的Message Handler、10G Ethernet MAC模块中的RX/TX路径处理逻辑）成为系统设计的标准需求。

本模块借鉴了工业标准设计，如：

1. X\_CAN Message Handler的高效FIFO缓存与DMA传输机制；
2. 10G Ethernet MAC中基于流式接口（Avalon-ST）进行帧解封装与CRC校验的高效数据处理策略；
3. HDLC控制器设计中的零填充/零去填充技术以确保帧同步；
4. USB Function IP Core中数据包解析与多通道输出管理模式。

本模块设计具备以下特点：

1. 支持固定格式与可变格式的帧结构检测；
2. 内置CRC校验功能，实现帧级数据完整性验证；
3. 采用异步FIFO缓存机制，实现不同时钟域下的稳定数据存取；
4. 支持格雷码编码处理，便于后续高速串行输出接口的数据同步；
5. 具备快速恢复机制，能够在检测到错误帧后及时恢复到下一帧检测；
6. 支持多通道输出，结合独立通道有效标志位，有效适配不同下游处理需求。

本模块既可独立使用，也可作为通信接口子系统（如Ethernet MAC、CAN控制器、PCIe接收端等）的基础组件。通过高度可配置化的设计，模块能够根据实际应用需求灵活调整参数，包括帧长、通道数量、FIFO深度、编码规则等。

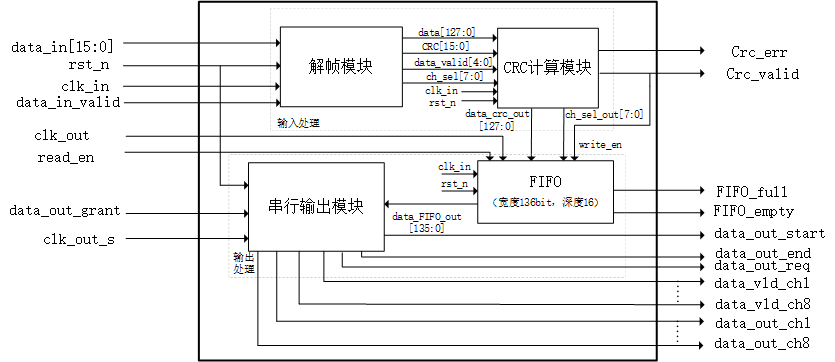
通过本模块的应用，系统能够实现高速、可靠、低延迟的数据接收与分发，有效提升整体数据链路处理能力，为多种高性能系统设计提供强有力的支持。

通过本模块的应用，系统能够实现高速、可靠、低延迟的数据接收与分发，有效提升整体数据链路处理能力，为多种高性能系统设计提供强有力的支持。

## 2.模块描述：

### 2.1功能描述

帧格式序列检测生成模块旨在对高速输入的数据流进行实时的帧格式识别、数据提取、完整性校验与多通道并行输出。模块整体架构遵循分层解耦设计，如图所示：



该模块完成如下功能：

1.解帧模块：解帧模块又分为三个子模块

（1）输入数据检测：在50MHz至100MHz范围内，正确采样并检测输入的数据流；

（2）帧格式识别：检测特定格式帧（帧头、通道选择、数据、CRC校验、帧尾），支持可变长度数据（16-128 bits);

（3）解帧与数据提取： 正确分离帧头/通道选择/帆尾，将数据部分抓取出来，并进行CRC校验；校验失败时，报告错误并忽略错误帧。

2.CRC计算模块：根据收到的数据计算理想CRC值，并将其与接收到的CRC做比较，如果一致说明数据有效，将数据发送给FIFO，反之则丢弃数据，同时根据要求输出crc\_err和crc\_valid信号。

3.异步FIFO缓存： 接收数据通过FIFO缓存，并将其输出给串行输出模块。支持异步读写，并提供FIFO空/满状态信号。

4.串行输出：将FIFO输出数据的数据位和通道选择信号分开，将数据转换为格雷码，然后根据通道选择信号，输出对应的通道有效使能，并讲数据从对应通道单比特串行输出。（该模块输出时钟clk\_out\_s为clk\_in的16倍频）

### 2.2测试要求

1. 功能测试：

每个功能特性（帧别，CRC校验，FIFO读写，数据编码）必须独立测试，确保功能完整。

1. 时序测试：

在50MHz-100MHz范围内测试模块效能，确保不同频率下正确工作。

1. 错误处理测试：

检测到输入数据错误时，模块应能正确处理并恢复正常工作。错误包括以下几种情况：

1. 检测到帧头及后续数据，但是没有收到帧尾，需丢弃当前帧。
2. 通道选择数据不正确（非独热码选择 1-8），需丢弃当前帧。
3. 接收数据长度超出范围（16~128bit），需丢弃当前帧。
4. 检测到 CRC 校验失败时，模块应能报告错误并丢弃当前错误数据。

在以上检测到输入错误/异常情况下，模块对当前错误帧全部数据进行丢弃，不进行编码输出，对下一帧数据要求能够正常输出。

1. 复位测试：

复位信号激活后，状态机和输出信号应恢复到初始状态。

复位后的恢复时间应符合设计要求（例如，不超过 `T` 个时钟周期）。

1. 仿真和要求：

使用 VCS 仿真工具进行全面功能与时序验证，要求代码覆盖95%以上，确保所有应性功能都有相应测试用例。

1. 功能完整性测试：

验证模块在正常数据流输入、异常输入（错误帧、CRC错误、同步丢失）情况下，均能正确检测、提取、校验并输出有效数据。

1. 边界条件测试：

包括最短帧、最长帧、高负载FIFO写满、连续错误帧恢复等边界情况。

1. 速率适应测试：

在输入时钟50-100MHz、输出时钟50MHz及串行时钟800MHz条件下，验证模块在不同频率组合下的稳定工作能力。

1. 异常恢复能力测试：

验证在遇到异常数据流时，模块能在5个clk\_in周期内快速恢复同步并进入下一帧检测。

1. 时序约束验证：

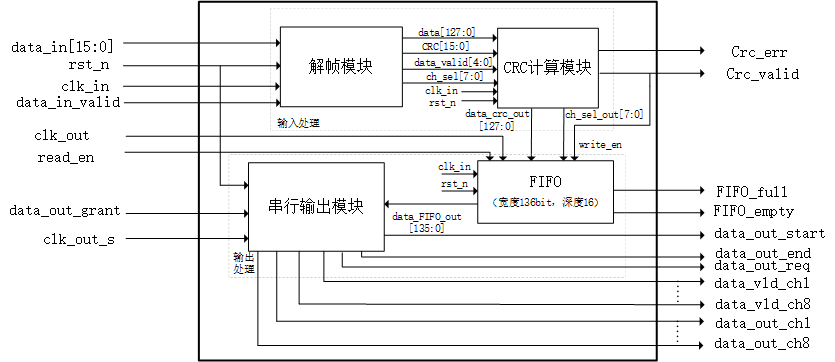
应通过静态时序分析（STA）工具验证关键时序路径，确保满足芯片设计要求。

### 2.3 关键性能指标

| **指标项** | **要求** |
| --- | --- |
| 帧检测准确率 | 100% |
| CRC校验准确率 | 100% |
| FIFO异步跨时钟可靠性 | 零数据丢失 |
| 支持最大帧长度 | 64 KB |
| 支持最大吞吐率 | 输入通道最高达1.6Gbps（16bit@100MHz），串行输出800Mbps/通道 |
| 错误恢复时间 | ≤ 5个clk\_in周期 |
| 支持通道数量 | 8路并行 |
| 支持帧格式 | 固定格式与可变格式，支持扩展 |

## 3.接口定义：

### 3.1顶层接口图



### 3.2端口信号列表

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Direction | Width(bits) | Description |
| data\_in | input | 16 | 待检测的输入数据流 |
| ret\_n | input | 1 | 复位模块，初始化状态 |
| clk\_in | input | 1 | 提供模块输入数据采样所需的时钟脉冲 |
| data\_in\_valid | input | 1 | 指示有效输入数据 |
| read\_en | input | 1 | 异步FIFO读使能 |
| clk\_out | input | 1 | 提供异步 FIFO 读取数据所需的时钟脉冲 |
| clk\_out\_s | input | 1 | 提供给串行输出所需的时钟脉冲 |
| data\_out\_grant | input | 1 | 允许数据输出信号 |
| crc\_err | output | 1 | 指示 CRC 校验结果错误 |
| crc\_valid | output | 1 | 指示 CRC 校验结果 |
| FIFO\_full | output | 1 | 指示 FIFO 是否已满 |
| FIFO\_empty | output | 1 | 指示 FIFO 是否为空 |
| data\_vld\_ch[1-8] | output | 1 | 数据有效信号通道 |
| data\_out\_ch[1-8] | output | 1 | 数据串行输出通道[1-8] |
| data\_out\_req | output | 1 | 数据发送请求信号 |
| data\_out\_start | output | 1 | 数据发送起始信号 |
| data\_out\_end | output | 1 | 数据发送结束信号 |

## 4.设计要求：

### 4.1解帧模块：

#### 4.1.1功能描述:

存储输入数据流，并能够正确识别帧头和帧尾,将输入数据流拆解成data、CRC、ch\_sel三个部分并将其发送给CRC计算模块。

具体功能介绍如下：

正常输入：将数据正确解帧并输出

错误输入：帧格式不满足要求时丢弃该帧

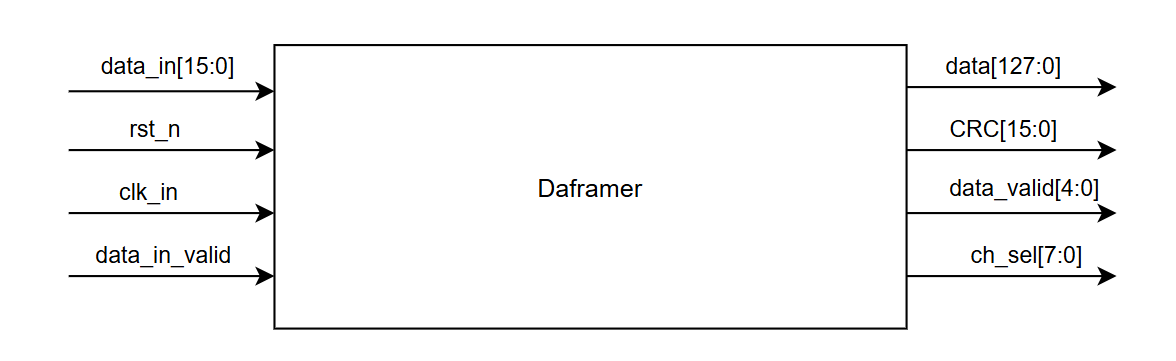
在一些边界情况下设计的行为：输入帧最大长度为128+88位时，接收输入帧需要28个clk\_in周期。

#### 4.1.2测试要求：

* 1. 输入数据应在不同的时钟周期内正确采样。
  2. 在不同输入频率下，输入数据的正确性应保持一致。
  3. 模块应能检测特定的帧格式，正确识别帧头、通道选择和帧尾，提取帧中的数据部分。
  4. 模块应能在连续输入不同的目标帧时，正确检测每一个目标帧。

#### 4.1.3模块架构与端口列表：

模块架构：



端口列表：

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Direction | Width(bits) | Description |
| clk\_in | input | 1 | 提供模块输入数据采样所需的时钟脉冲 |
| rst\_n | input | 1 | 复位模块，初始化状态 |
| data\_in | input | 16 | 待检测的输入数据流 |
| data\_in\_valid | input | 1 |  |
| data | output | 128 | 输入帧中的数据部分 |
| ch\_sel | output | 8 | 输入帧中的通道选择信号 |
| CRC | output | 16 | 输入帧中的CRC校验码 |
| data\_valid | output | 5 |  |

#### 4.1.4信号接口描述：

* 输入：

1. clk\_in

描述：提供模块输入数据采样所需的时钟脉冲。

作用：驱动数据采样和帧格式检测。

指标：时钟频率50MHz。

1. rst\_n

描述： 复位模块，初始化状态。

作用： 在检测到异常或需要重新开始时复位模块。

指标： 异步复位，低电平有效

1. data\_in[15:0]

描述： 待检测的输入数据流。

作用： 提供要检测的帧格式数据。

指标： 16 位位宽，按照Big-Endian方式输入。

1. data\_in\_valid

描述：输入数据有效标志

作用：指示数据的有效输入。

指标：1比特

* 输出:

1. data[127:0]

描述： 输入帧中的数据部分。

作用： 提供给后续模块进行CRC计算、验证以及输出。

指标： 128位位宽，若实际数据位宽小于该位宽则高位补0。

1. ch\_sel[7:0]

描述： 输入帧中的通道选择信号。

作用： 提供给后续模块进行输出通道选择。

指标： 8位位宽，格式为独热码。

1. CRC[15:0]

描述： 输入帧中的CRC校验码。

作用： 提供给后续模块进行CRC校验。

指标： 32位位宽。

1. data \_valid[4:0]

描述：指示解帧模块输出数据有效位标志。

作用：提供给CRC模块来指明输出数据有效位。

指标：4比特。

#### 4.1.5逻辑功能设计：

#### 一．正常帧格式

#### 正常帧格式流程

#### 完整工作流程为：IDLE→HEAD\_1→HEAD\_2→CH\_SEL→DATA→TAIL\_1→TAIL\_2→DONE→IDLE

#### 系统通过连续接收两个E0E0帧头（HEAD\_1和HEAD\_2阶段）启动流程

#### 在CH\_SEL状态检测到有效的独热码后进入数据接收阶段

#### DATA状态持续累加data\_valid直到遇到结束标志0e0e

#### 最后通过两级结束确认（TAIL\_1和TAIL\_2）完成完整帧接收

#### 

#### 数据缓存

#### 在DATA状态通过data\_valid计数器记录有效数据长度

#### 每次接收非结束帧（≠0e0e）时计数器+2（16bit）

#### 该机制确保系统能准确判断数据段是否满足2-16字节的有效范围要求

#### 异常帧格式

#### 1.帧头错误

#### 

#### 异常路径：IDLE→HEAD\_1→IDLE

#### 当第二个帧头不符合e0e0要求时（图示为e0e1）系统立即返回IDLE状态

#### 这种严格校验机制防止错误帧头导致的误操作

#### 通道选择错误

#### 

#### 异常路径：IDLE→HEAD\_1→HEAD\_2->IDLE

#### 通道选择数据需要满足奇校验要求（高8位异或结果为1）;任何校验失败都会立即返回IDLE状态，保证可靠性

#### 3.数据位不足错误

#### 

#### 数据位错误（小于16bit)

#### 异常路径：IDLE→HEAD\_1→HEAD\_2→CH\_SEL→DATA->TAIL\_1→TAIL\_2→IDLE

#### 当data\_valid<2字节时（图示仅1字节）系统在TAIL\_2状态检测到数据长度不足；直接复位到IDLE状态放弃当前帧

#### 4.数据位超限错误

#### 

#### 异常路径：IDLE→HEAD\_1→HEAD\_2→CH\_SEL→DATA->TAIL\_1→TAIL\_2→IDLE

#### 当data\_valid>16字节时（图示超限;同样在TAIL\_2状态触发错误处理

#### 系统通过复位避免处理异常数据

#### 4.超时处理机制

#### 

#### 超时处理

#### 在任何状态（特别是DATA状态）发生超时系统(超时5个时钟周期）都会强制跳转至IDLE状态；该机制保证通信异常时能及时恢复初始状态.

图 1:状态机循环图

|  |  |  |
| --- | --- | --- |
| 序号 | 状态 | 描述 |
| 1 | IDLE | 当data\_in = 16’he0e0跳转 HEAD\_1 ；否则维持IDLE |
| 2 | HEAD\_1 | 当data\_in = 16’e0e0 跳转 HEAD\_2: 否则跳转IDLE |
| 3 | HEAD\_2 | 当data\_in[15:8]为独热码时，跳转 CH\_SEL ；否则跳转IDLE |
| 4 | CH\_SEL | 当data\_in != 16’h0e0e时，data\_valid = data\_valid + 2 ，跳转DATA；当data\_in = 16’he0e0时,data\_valid = 0;跳转TAIL\_1； |
| 5 | DATA | 当data\_in ! =16’h0e0e时，data\_valid = data\_valid +2,保持DATA ;当data\_in = 16’h0e0e时，跳转TAIL\_1； |
| 6 | TAIL\_1 | 当data\_in !=16’h0e0e时,data\_in != 16’h0e0e时，跳转DATA；当 data\_in = 16’h0e0e时，跳转TAIL\_2 |
| 7 | TAIL\_2 | 当2大于等于data\_valid<16时,跳转DONE；当data\_valid>16或者data\_valid<2时，跳转IDLE |
| 8 | DONE | 跳转进入IDLE |

### 4.2 CRC计算模块：

#### 4.2.1功能描述：

根据收到的数据计算理想CRC值，并将其与接收到的CRC做比较，如果一致说明数据有效，将数据发送给FIFO，反之则丢弃数据，同时根据要求输出Crc\_err和Crc\_valid信号。

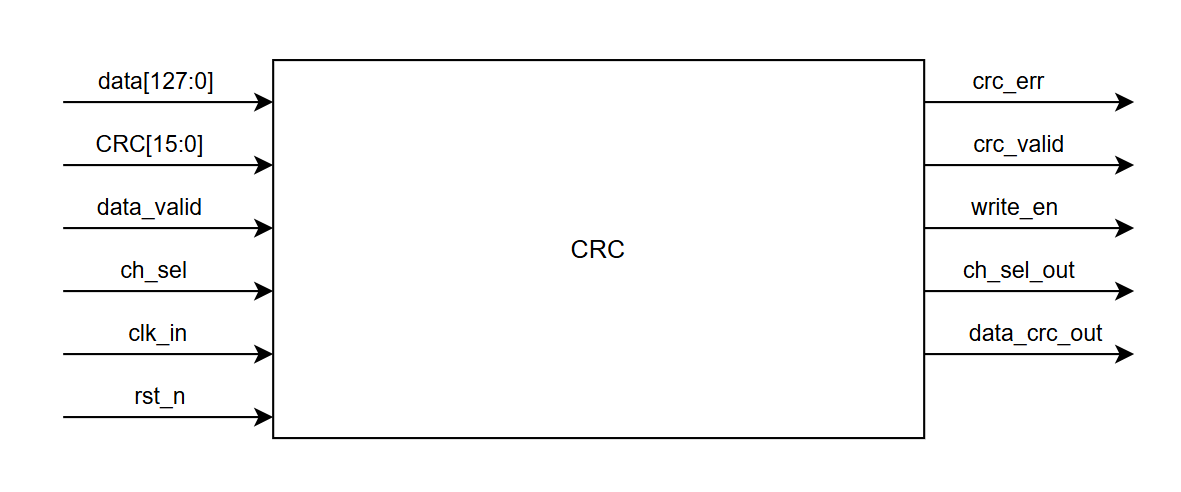
* 具体功能介绍如下：

1. 正常输入：CRC校验通过，将数据存入fifo
2. 错误输入：CRC校验不通过时输出CRC\_err信号。

#### 4.2.2测试要求：

1. 进行 CRC 校验，确保数据完整性。
2. 若 CRC 校验失败，模块应能正确处理并报告错误，不进行编码输出。

#### 4.2.3模块架构与端口列表

* 模块架构
* 端口列表

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Direction | Width(bits) | Description |
| clk | input | 1 | 时钟信号 |
| data | input | 128 | 输入数据 |
| data\_valid | input | 5 | 输入数据有效字节数 |
| CRC | input | 16 | 输入CRC校验码 |
| rst\_n | input | 1 | 复位信号 |
| data\_crc\_out | output | 128 | 输出数据 |
| crc\_err | output | 1 | CRC校验失败信号 |
| crc\_valid | output | 1 | CRC校验成功信号 |
| ch\_sel | output | 8 | 输入帧通道选择信号 |
| write\_en | output | 1 | fifo写使能 |

#### 4.2.3信号接口描述：

* 输入：

1. clk\_in

描述：提供模块输入数据采样所需的时钟脉冲。

作用：驱动数据采样和帧格式检测。

指标：时钟频率50MHz。

1. rst\_n

描述： 复位模块，初始化状态。

作用： 在检测到异常或需要重新开始时复位模块。

指标： 异步复位，低电平有效。

1. data[127:0]

描述： 输入帧中的数据部分。

作用： 进行CRC计算，若CRC校验正确，则输出给后续模块。

指标： 128位位宽，若实际数据位宽小于该位宽则高位补0。

1. data\_valid[4:0]

描述： 描述输入数据有效字节数。

作用： 表示输入数据的有效字节数。

指标： 五位位宽。

1. CRC[15:0]

描述： 输入帧中的CRC校验码。

作用： 进行CRC校验。

指标： 16位位宽。

* 输出：

1. crc\_valid

描述： 指示CRC校验结果正确。

作用: 在CRC校验成功时激活。

指标: 输出为高电平有效，数据CRC校验正确输出周期内拉高。

1. crc\_err

描述: 指示CRC校验结果错误。

作用: 在CRC校验错误时激活。

指标: 输出为高电平有效，数据CRC校验错误周期内拉高。

1. data\_crc\_out[127:0]

描述： 输入帧中的数据部分。

作用： CRC校验正确时输出给后续模块。

指标： 128位位宽，若实际数据位宽小于该位宽则高位补0。

1. write\_en

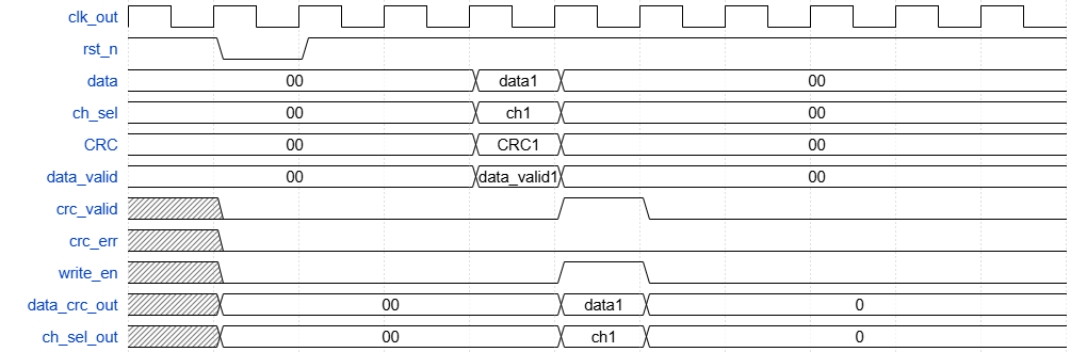
描述： FIFO写使能。

作用： CRC校验正确时，允许FIFO写入一次数据。

指标： 高电平有效。

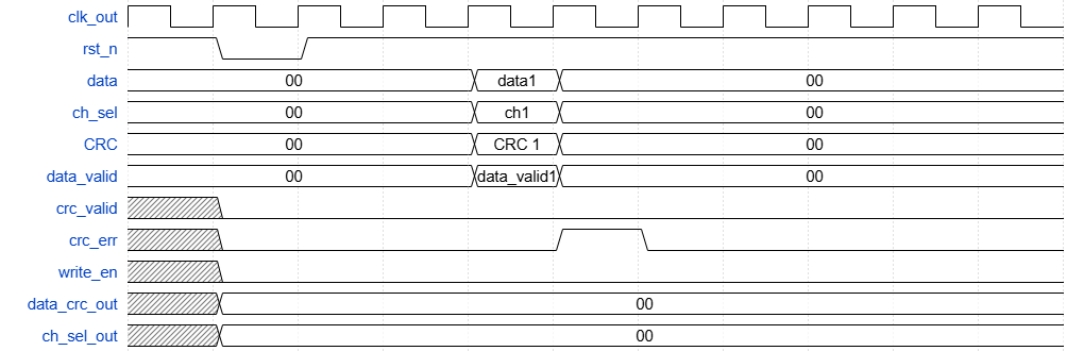
#### 4.2.4逻辑功能设计

CRC校验正确时序图：



CRC校验正确时，在输入有效数据的下一拍将data和ch\_sel信号传递给下级模块，同时输出crc\_valid为高，所有有效输出保持一个时钟周期后归零。

CRC校验异常时序图：



CRC校验异常时，在输入有效数据的下一拍输出crc\_err为高，持续一个时钟周期后归零，同时所有其他输出保持为0。

**CRC校验算法**：该模块使用CRC-16-CCITT算法，多项式为x16 + x12 + x5 + 1。

具体步骤如下：

（1）预置1个16位的寄存器为十六进制FFFF（即全为1），称此寄存器为CRC寄存器；

（2）将输入数据按字节进行拆解。把第一个8位二进制数据（既输入数据的第一个字节）与16位的CRC寄存器的高8位相异或，把结果放于CRC寄存器，低8位数据不变；

（3）把CRC寄存器的内容左移一位，用0填补最低位，并检查左移后的移出位；

（4）如果移出位为0：重复第3步（再次左移一位）；如果移出位为1，CRC寄存器与多 项式1021（对应的二进制码为1 0000 0010 0001）进行异或；

（5）重复步骤3和4，直到右移8次，这样整个8位数据全部进行了处理；

（6）重复步骤2到步骤5，进行输入数据的下一个字节的处理；

（7）将输入数据所有字节按上述步骤计算完成后，最后得到的CRC寄存器内容即为CRC码。

## 4.3 FIFO模块：

#### 4.3.1功能描述：

在写使能信号下，将来自CRC计算模块输出的八位通道选择信号和128位有效数据拼接成136位数据寄存在同一地址空间内。在读使能信号下，将136位数据输出到串行输出模块。

* 具体功能介绍如下：

1. 可对FIFO深度，宽度进行参数化定义。
2. 能够实现数据的异步读写功能，且读出的数据是先入先出的顺序。
3. 能够指示FIFO空、满状态。
4. 能够实现将输入的通道选择信号和数据拼接起来，并寄存在同一地址空间。

在错误和边界情形下设计的行为：

* 错误输入：

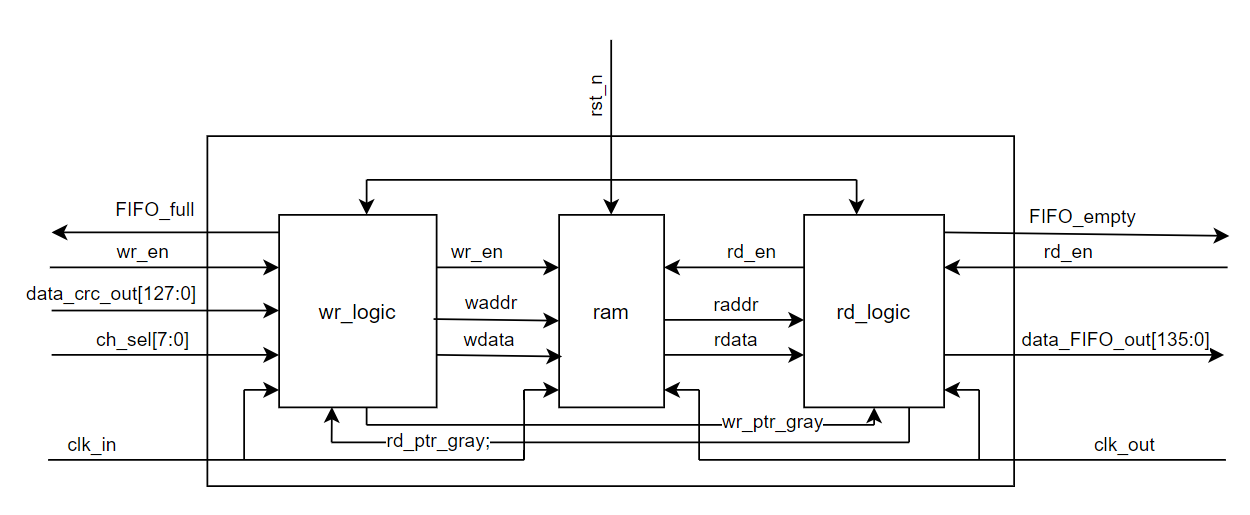
1. 在FIFO为空时读数据，则会读出全0。
2. 在FIFO为满时写数据将会无法写入。

* 边界情形：
  + - 1. 当读指针为0，在连续写入16个数据时，FIFO\_full信号会立刻拉高，提示FIFO已写满。
      2. 当写指针不变，连续读出16个数据时，FIFO\_empty信号会立刻拉高，提示FIFO已读空。
      3. 在异步读写情形下，会存在虚满和虚空的情形。

#### 4.3.2测试要求：

1. FIFO 应能正确缓存提取的数据。
2. 提供 FIFO 空（fifo\_empty）和 FIFO 满（fifo\_full）信号状态指示。
3. FIFO 应支持异步读写操作。

#### 4.3.3模块架构与端口列表

* 模块架构：
* 端口列表：

FIFO端口列表：

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Direction | Width(bits) | Description |
| clk\_in | input | 1 | FIFO写时钟 |
| clk\_out | input | 1 | FIFO读时钟 |
| rst\_n | input | 1 | 复位信号，低电平有效 |
| data\_crc\_out | input | 128 | crc模块输出的有效数据 |
| ch\_sel | input | 8 | 通道选择信号 |
| wr\_en | input | 1 | 写使能信号 |
| rd\_en | input | 1 | 读使能信号 |
| data\_FIFO\_out | output | 136 | FIFO输出数据 |
| FIFO\_full | output | 1 | FIFO满信号 |
| FIFO\_empty | output | 1 | FIFO空信号 |

FIFO配置参数列表：

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | units | Width | Description |
| DATA\_WIDTH | bit | 136 | FIFO的数据宽度 |
| DATA\_DEPTH | bit | 16 | FIFO的深度 |
| PTR\_WIDTH | bit | 5 | 指针位宽 |

FIFO内部信号列表：

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Data Type | Width(bits) | Description |
| wr\_ptr\_reg | reg | 5 | 二进制写指针 |
| rd\_ptr\_reg | reg | 5 | 二进制读指针 |
| rd\_ptr\_gray\_sync1 | reg | 5 | 在写时钟域同步的读指针第一拍 |
| rd\_ptr\_gray\_sync2 | reg | 5 | 在写时钟域同步的读指针第二拍 |
| wr\_ptr\_gray\_sync1 | reg | 5 | 在读时钟域同步的写指针第一拍 |
| wr\_ptr\_gray\_sync2 | reg | 5 | 在读时钟域同步的写指针第二拍 |
| wr\_ptr\_gray | wire | 5 | 格雷码写指针 |
| rd\_ptr\_gray | wire | 5 | 格雷码读指针 |
| wr\_ptr\_ture | wire | 4 | ram的写地址 |
| rd\_ptr\_true | wire | 4 | ram的读地址 |

#### 4.3.4信号接口描述：

* 输入：

1. clk\_in

描述：FIFO写时钟。

作用：驱动数据采样和帧格式检测。

指标：时钟频率为50MHz。

1. rst\_n

描述： 复位模块，初始化状态。

作用： 在检测到异常或需要重新开始时复位模块。

指标： 异步复位，低电平有效。

1. clk\_out

描述：FIFO读时钟。

作用：驱动FIFO读出数据。

指标：时钟频率50MHz，与clk\_in反相。

1. data\_crc\_out[135:0]

描述： 输入帧中的有效数据部分。

作用： CRC校验正确时写入FIFO。

指标： 136位位宽，若实际数据位宽小于该位宽则高位补0。

1. ch\_sel[7:0]

描述： 输入帧中的通道选择信号。

作用： 进行输出通道选择。

指标： 8位位宽，格式为独热码。

1. wr\_en

描述： FIFO写使能。

作用： 允许FIFO写入一次数据。

指标： 高电平有效。

1. rd\_en

描述： FIFO读使能。

作用： 允许FIFO读出一次数据。

指标： 高电平有效。

* 输出：

1. FIFO\_full

描述: 指示FIFO是否为空。

作用: 当FIFO为空时激活。

指标: 输出为高电平有效。

1. FIFO\_empty

描述: 指示FIFO是否已满。

作用: 当FIFO已满时激活。

指标: 输出为高电平有效。

1. data\_FIFO\_out[135:0]

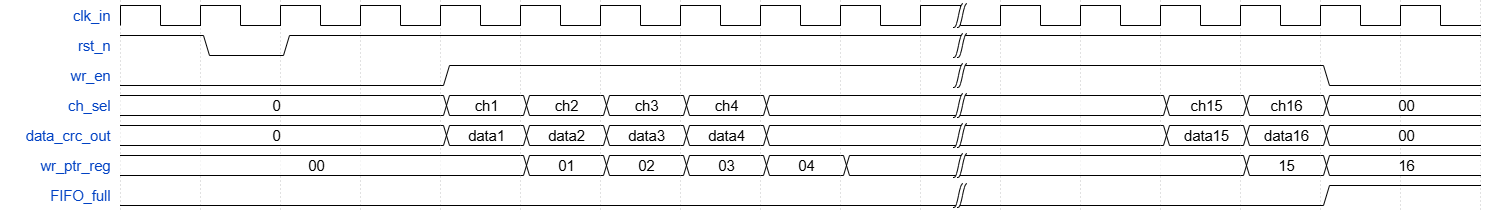
描述： 将通道使能信号和数据拼接存储在一起。

作用： FIFO读出该数据，给后级模块进行串行输出。

指标： 135位位宽，若实际数据位宽小于该位宽则高位补0。

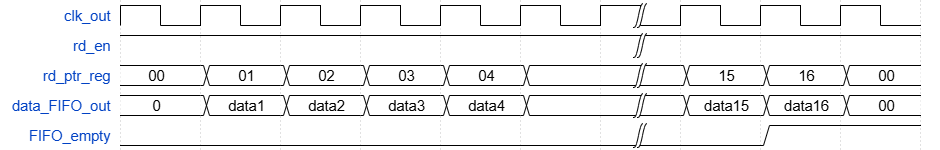
#### 4.3.5逻辑功能设计

写逻辑时序:

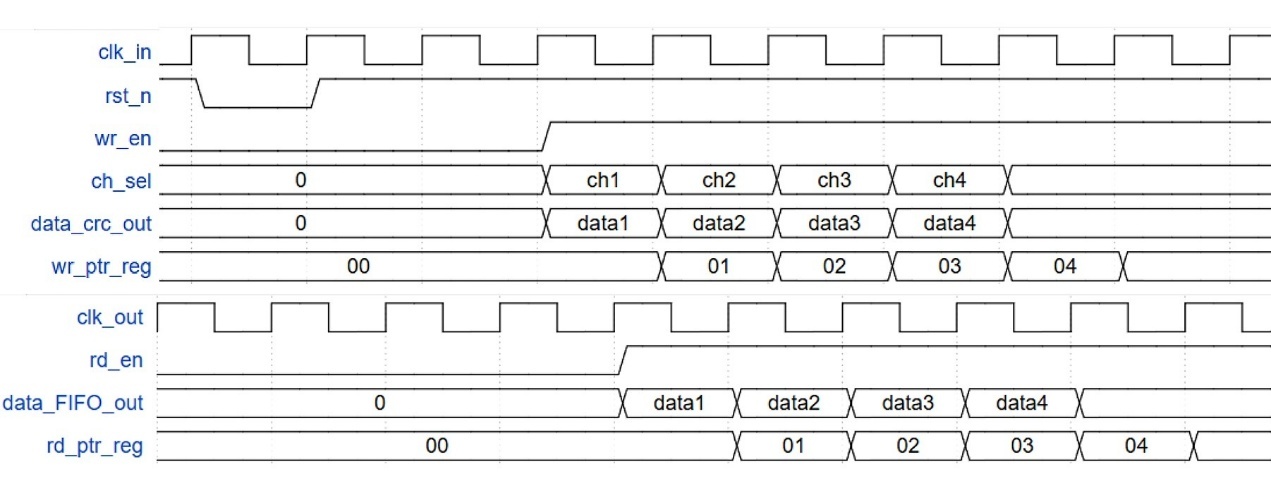


异步FIFO模块在复位信号(rst\_n)为低时，复位FIFO回到初始状态。在复位信号拉高后，在读使能(wr\_en)作用下，FIFO在时钟沿写入数据，本模块设计将CRC模块输出的通道选择信号(ch\_sel)和数据(data\_crc\_out)存储在FIFO的同一地址空间，目的是方便串行输出模块对数据检索输出。每写入一个数据，写指针(wr\_ptr\_reg)加一。FIFO\_full信号的设计逻辑是：首先将读，写指针转换为格雷码，能够避免指针跨时钟域的亚稳态问题。然后格雷码形式的读指针同步到写时钟域，通过打俩拍进行同步。当格雷码形式的读、写指针的最高位和次高位不同，其余位相同时，认为是写满，则拉高FIFO\_full信号。

读逻辑时序：



异步FIFO模块在读使能(rd\_en)作用下，在读时钟（clk\_out）边沿读出数据，每读出一个数据，读指针加一。FIFO\_empty信号的设计逻辑是: 首先将读，写指针转换为格雷码，能够避免指针跨时钟域的亚稳态问题。然后格雷码形式的写指针同步到读时钟域，通过打俩拍进行同步。当格雷码形式的读、写指针的最高位和次高位相同，其余位相同时，认为是读空，则拉高FIFO\_empty信号。

异步读写时序：

在本设计中，规范书要求读写时钟频率是同频异步，异步FIFO模块可以在写使能(wr\_en)作用下，在写时钟边沿写入数据，同时在读使能(rd\_en)作用下，在读时钟边沿读出数据。并且保证写入和读出的数据相同。

### 4.4串行输出模块：

#### 4.4.1功能描述：

将FIFO输出数据的数据位和通道选择信号分开，将数据转换为格雷码，然后根据通道选择信号，输出对应的通道有效使能，并讲数据从对应通道单比特串行输出。

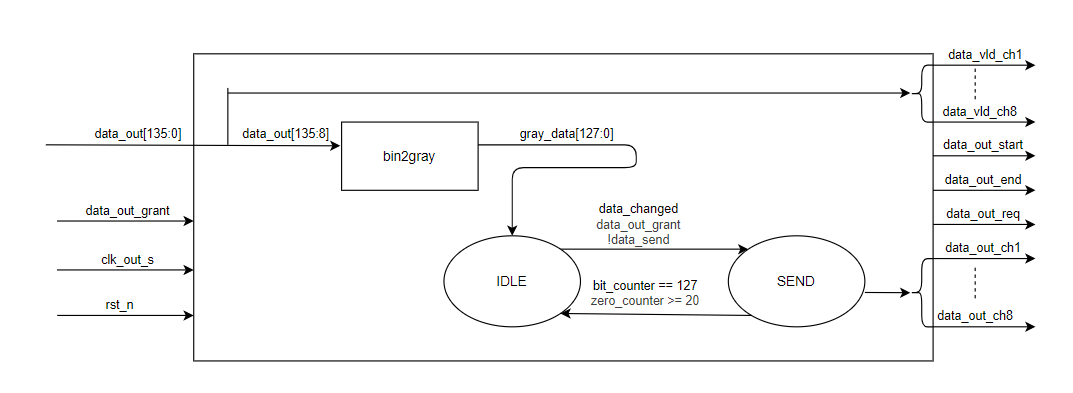
* 具体功能介绍如下：

1. 实现将FIFO输出的136bit数据分割，[7:0]为通道选择信号，[135:8]为数据位。
2. 实现将128比特数据每四位一组转换为格雷码。
3. 实现根据通道选择信号将对应的通道有效使能信号拉高。
4. 实现将转换为格雷码的数据在对应的输出通道单比特输出。

#### 4.4.2测试要求：

* + 1. 从 FIFO 中读取的数据应按照指定的格雷码编码方式进行正确编码。
    2. 编码后的数据应按顺序串行输出。
    3. 输出信号应在目标帧解帧并编码完成后正确激活。
    4. 输出信号在检测到目标帧并提取数据后的保持时间应符合设计要求

#### 4.4.3模块架构与端口列表

* 模块架构：
* 端口列表：

串行输出模块端口列表

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Direction | Width(bits) | Description |
| data\_out | input | 136 | 输入数据 |
| clk\_out\_s | input | 1 | 串行输出时钟 |
| rst\_n | input | 1 | 复位信号 |
| data\_out\_grant | input | 1 | 允许数据输出信号 |
| data\_vld\_ch[1-8] | output | 1 | 通道有效使能信号 |
| data\_out\_ch[1-8] | output | 1 | 输出通道信号 |
| data\_out\_req | output | 1 | 数据发送请求信号 |
| data\_out\_start | output | 1 | 数据发送起始信号 |
| data\_out\_end | output | 1 | 数据发送结束信号 |

状态机列表

|  |  |
| --- | --- |
| State | Description |
| IDLE | 空状态 |
| SEND | 发送数据 |

内部信号列表：

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Data Type | Width(bits) | Description |
| gray\_data | reg | 128 | 格雷码数据 |
| bit\_counter | reg | 7 | 记录发送的位数 |
| shift\_reg | reg | 128 | 寄存数据 |
| zero\_counter | reg | 5 | 记录连续零的个数 |
| data\_sent | reg | 1 | 标识是否发送完一组数据 |
| data\_changed | reg | 1 | 标识数据是否变化 |
| prev\_data\_out | reg | 136 | 寄存前一个数据 |

#### 4.4.4信号接口描述：

* 输入：

1. clk\_out\_s

描述：串行输出时钟。

作用：驱动数据串行输出。

指标：时钟频率为clk\_in的16倍。

1. rst\_n

描述： 复位模块，初始化状态。

作用： 在检测到异常或需要重新开始时复位模块。

指标： 异步复位，低电平有效。

1. data\_ out[135:0]

描述： FIFO模块读出的数据，进入串行输出模块。 data\_out[135:8]为数据位，data\_out[7:0]为通道选择信号。

作用：将该数据转换为格雷码后进行单比特串行输出。

指标：135位位宽，若实际数据位宽小于该位宽则高位补0。

1. data\_out\_grant

描述：允许数据输出信号。

作用：可以增加下级模块对输出通道的控制，并能增加输出效率。

指标：1位位宽，在接受到data\_out\_req信号后，由下级模块反馈是否允许数据输出。

* 输出：

1. data\_vld\_ch[1-8]

描述： 单比特通道有效使能信号。

作用： 与通道选择信号对应，指示哪个输出通道有效。

指标： 八个输出通道中只能同时有一个通道有效，高电平有效。

1. data\_out\_ch[1-8]

描述： 八个独立输出通道。

作用： 根据通道选择信号，在指定通道内串行输出经过格雷码编码后的数据。

指标： 连续单比特输出数据。

1. data\_out\_req

描述： 数据发送请求信号。

作用： 对下级模块发送数据请求发送信号，等到下级模块的反馈信号才能发送数据，增强下级模块的控制能力。

指标： 在进入SEND状态时，发送请求信号。

1. data\_out\_start

描述： 单比特数据开始发送起始标识。

作用： 指示数据开始发送。

指标： 在数据开始发送时置1，维持一个周期后置0。

1. data\_out\_end

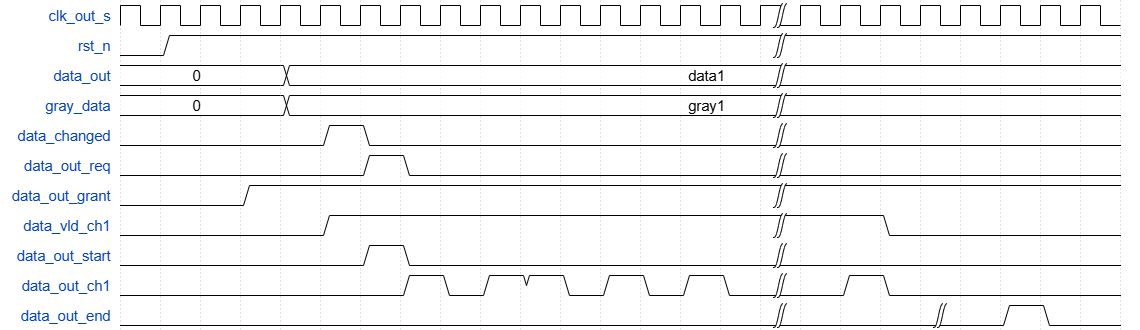
描述： 单比特数据开始发送结束标识。

作用： 指示数据发送结束。

指标： 在数据发送结束时置1，只至有下一个数据进入时置0。

#### 4.4.5逻辑设计

时序图：



以通道1为例，串行输出模块首先分割输入数据data\_out[135:0],其中data\_out[7:0]为通道选择信号（独热码），根据其信号来指示数据有效信号通道data\_out\_ch[1-8]，将其信号拉高。data\_out[135:8]为数据位，通过bin2gray函数将二进制数据每四位一组转换为格雷码。在本模块定义了IDLE,SEND俩个状态，在IDLE状态将格雷码数据存入寄存器。data\_changed信号会在时钟沿检测是否有新数据输入，当有新数据输入后，拉高一个周期。SEND状态为数据发送状态，当data\_changed信号拉高，状态机由IDLE进入SEND，开始发送数据，data\_out\_start信号拉高一个周期，标识数据发送起始位。由于在本系统中为数据存取方便，将解帧模块的数据不满128bit的数据进行了高位补零。本模块为提高输出效率，增加data\_out\_end信号，当128bit数据完全输出，或者在数据输出中检测到连续二十个零则拉高data\_out\_end信号，标识输出完毕，可以输入新的数据。