FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Implementação em FPGA de um conversor HDMI para transmissão ótica em série

Ana Marisa Oliveira Barbosa

VERSÃO DE TRABALHO

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador: Prof. Doutor João Paulo de Castro Canas Ferreira

Co-orientador: Prof. Dr. Henrique Manuel de Castro Faria Salgado

Supervisor Externo: Dr. Luís Manuel de Sousa Pessoa

7 de Abril de 2017

Resumo

Este documento contempla o trabalho realizado para problema proposto neste projeto no âmbito da Dissertação de Mestrado. Este relatório contém uma análise introdutória do problema, a revisão bibliográfica do mesmo, de seguida é realizada a caracterização do problema e a proposta de resolução do mesmo. Por fim, é ainda realizado uma calendarização das tarefas a serem realizadas e metodologias de abordagem das mesmas.

Abstract

Now I have to write it in English

Agradecimentos

Agradecer a toda a gente no mundo.

Marisa Oliveira

'The journey is the reward'

Steve Jobs

Conteúdo

1 Introdução **1**
- 1.1 Enquadramento Geral . . . 1
- 1.2 Motivação . . . 2
- 1.3 Objetivos . . . 4
- 1.4 Estrutura da Dissertação . . . 4

2 Revisão Bibliográfica **5**
- 2.1 Interfaces de transmissão de video/audio . . . 5
- 2.2 HDMI (*High Definition Multimedia Interface*) . . . 6
  - 2.2.1 DDC - *Display Data Channel* . . . 6
  - 2.2.2 TMDS - *Transition-Minimized Differential Signaling* . . . 6
  - 2.2.3 CEC - *Consumer Electronics Control* . . . 7
  - 2.2.4 ARC - *Audio Return Channel* . . . 7
  - 2.2.5 HEC - *HDMI Ethernet Channel* . . . 7
- 2.3 HDMI implementado sobre a FPGA . . . 7
  - 2.3.1 Conexão à FPGA XILINX VC7203 Virtex-7 . . . 8
  - 2.3.2 Transmissor e Recetor . . . 9
- 2.4 Conexão de alta velocidade em série . . . 14
  - 2.4.1 Arquitetura de serializador e deserializador . . . 15
  - 2.4.2 Considerações na implementação deste tipo de arquitetura . . . 16
  - 2.4.3 Serializador e Deserializador disponíveis na FPGA . . . 19

3 Capítulo Exemplo **35**
- 3.1 Introdução . . . 35
- 3.2 Secção Exemplo . . . 35
  - 3.2.1 Exemplo de Figura . . . 36
  - 3.2.2 Exemplo de Tabela . . . 36
- 3.3 Secção Exemplo . . . 37
- 3.4 Resumo . . . 37

4 Mais um Capítulo **39**
- 4.1 Secção Exemplo . . . 39
- 4.2 Mais uma Secção . . . 40
- 4.3 Resumo ou Conclusões . . . 41

5 Conclusões e Trabalho Futuro **43**
- 5.1 Satisfação dos Objectivos . . . 43
- 5.2 Trabalho Futuro . . . 44

A Loren Ipsum **45**
A.1 O que é o *Loren Ipsum*? 45
A.2 De onde Vem o Loren? 45
A.3 Porque se usa o Loren? 46
A.4 Onde se Podem Encontrar Exemplos? 46

Lista de Figuras

2.1 Vista Geral da FPGA VC7203 Virtex-7 retirada de [1] 8
2.2 Diagrama de blocos de TB-FMCH-HDMI2 RX retirado de [2] 10
2.3 Amostragem dos dados provenientes da FPGA no recetor, retirada de [2] 11
2.4 TB-FMCH-HDMI2 RX, retirada de [2] 11
2.5 Diagrama de blocos de TB-FMCH-HDMI2 TX retirado de [2] 12
2.6 Amostragem dos dados provenientes do FMC no recetor, retirada de [2] 12
2.7 TB-FMCH-HDMI2 TX, retirada de [2] 13
2.8 Configuração DDC normal, retirada de [2] 14
2.9 Configuração DDC "*through*", retirada de [2] 15
2.10 Arquitetura simples de um ser/des, retirada de [3] 16
2.11 Arquitetura de PISO/SIPO, retirada de [3] 18
2.12 Identificação dos transcetores GTX na FPGA VC7203 Virtex-7, retirada de [1] . 20
2.13 Arquitetura geral dos transcetores GTX, retirada de [4] 21
2.14 Diagrama de blocos de um transmissor GTX, retirada de [5] 21
2.15 Diagrama de blocos de um recetor GTX, retirada de [5] 25
2.16 Equalizador em modo LPM, retirada de [5] 26
2.17 Equalizador em modo DFE, retirada de [5] 27
2.18 Detalhes do circuito CDR (*Clock data recovery*), retirada de [5] 27
2.19 Mecanismo de obtenção da "vírgula", retirado de [5] 29
2.20 Mecanismo de obtenção da "vírgula" quando ALIGN_COMMA_DOUBLE=1, retirado de [5] 30

Lista de Tabelas

2.1 Nomes dos pins da interface FMC de TB-FMCH-HDMI2 RX, adaptada de [2] . . 10
2.2 Nomes dos pins da interface FMC de TB-FMCH-HDMI2 TX, adaptada de [2] . . 13
2.3 Configuração do tamanho dos dados de TXDATA, adaptada de [5] 22
2.4 Configuração da frequência de TXUSRCLK2, adaptada de [5] 23
2.5 Configuração do tamanho dos dados de RXDATA, adaptada de [5] 32
2.6 Configuração da frequência de TXUSRCLK2, adaptada de [5] 33

3.1 Tabela Exemplo 36

Abreviaturas e Símbolos

ARC	*Audio Return Channel*
BER	*Bit Error Rate*
CDR	*Clock Data Recovery*
CEC	*Consumer Electronics Control*
CMU	*Clock Multiplier Unit*
DDC	*Display Data Channel*
DFE	*Decision Feedback Equalizer*
DVI	*Digital Video Interface*
EDID	*Extended Display Identification Channel*
EEPROM	*Electrically erasable programmable read-only memory*
EIA/CEA	*Electronic Industry Alliance/ Consumer Eletronics Association*
FEC	*Forward Error Correction*
FEUP	Faculdade de Engenharia da Universidade do Porto
FIFO	*First-In First-Out*
FMC	*FPGA Mezzanine Cards*
FPGA	*Field-Programmable Gate Array*
HDCP	*High-bandwith Digital Content Protection*
HDMI	*High Definition Multimedia Interface*
HDTV	*High-Definition television*
HEC	*HDMI Ethernet Channel*
HPC	*High Pin Count*
iBrow	*Innovative ultra-BROadband ubiquitous Wireless communications through terahertz transceivers*
INESC-TEC	Instituto de Nacional de Engenharia de Sistema e Computadores Tecnologias e Ciências
LPCM	*Linear Pulse Code Modulation*
MIMO	*Multiple Input Multiple Output*
PCS	*Physical Coding Sublayer*
PISO	*Parallel-Input Serial-Output*
PLL	*Phase-Locked Loop*
PMA	*Physical Medium Attachment Sublayer*
PRBS	*Pseudo Random Bit Sequence*
QAM	*Quadrature Amplitude Modulation*
RTD	*Resonant Tunneling Diode*
SIPO	*Serial-Input Parallel-Output*
TMDS	*Transition- Minimized Differential Signaling*
VESA	*Video Electronics Standards Association*

Capítulo 1

Introdução

Este trabalho surge no contexto da unidade curricular Preparação para a Dissertação, pertencente ao plano de estudos do Mestrado Integrado em Engenharia Eletrotécnica e de Computadores, sendo que esta mesma unidade curricular dá início ao trabalho a ser realizado no semestre seguinte na unidade curricular Dissertação.

1.1 Enquadramento Geral

Ao longo das últimas décadas a sociedade tem vindo a tornar-se cada vez mais dependente das comunicações com e sem fios, não só em termos empresariais, mas também em termos pessoais. Esta tendência tem vindo a vincar-se recentemente, com a crescente utilização de tablets e smartphones, tornando os recursos atuais incapazes de responder a tal procura. E cada vez esta exigência irá aumentar prevendo-se a necessidade de ligações na ordem das centenas de Gb/s no ano de 2020, essencialmente para comunicações a curta distância. Daqui conclui-se que os recursos que existem atualmente não são capazes de responder a esta necessidade crescente de comunicações de alto débito, e como tal é necessário urgentemente o desenvolvimento de tecnologias não só capazes de satisfazer esta procura, mas ao mesmo tempo que o façam de forma eficiente em termos energéticos e financeiros. Neste contexto enquadra-se o projeto iBrow (Innovative ultra-BROadband ubiquitous Wireless communications through terahertz transceivers), o qual está a ser parcialmente desenvolvido pela equipa de investigação de tecnologias óticas e eletrónicas do INESC-TEC, que vem responder a esta necessidade de uma forma eficiente.

Este projeto vem propor o desenvolvimento de uma tecnologia capaz de responder a esta necessidade de comunicações de alto débito através de uma utilização eficaz do espetro de frequências, promovendo a utilização de bandas de frequência mais altas, desde 60 GHz até 1 THz. Para além disso vem também propor uma metodologia, que pela primeira vez permite um baixo custo de manufaturação de transcetores capazes de atingir altos débitos de transmissão para que possam ser perfeitamente integrados em redes de comunicações ótica de grande velocidade.

Toda esta crescente de consumo por parte dos utilizadores de novas e cada vez mais tecnologias não se verifica apenas na necessidade de aumento de largura de banda para as comunicações, mas

existe também uma necessidade extrema da existência de interfaces digitais de vídeo e som que não só sejam capazes de fazer chegar ao utilizador sinais de alto débito, mas que ao mesmo tempo o façam de maneira segura no sentido de proteger eventuais cópias não autorizadas. Assim sendo, o desenvolvimento de um conversor HDMI (High Definition Multimedia Interface) de alto débito enquadra-se perfeitamente nesta necessidade sendo que é a interface de vídeo e áudio standard e que implementa o protocolo HDCP (High-bandwith Digital Content Protection) que protege a reprodução de sinais em dispositivos não autorizados.

Existem várias interfaces digitais que implementam o protocolo referido anteriormente, entre elas destacam-se DisplayPort, DVI e HDMI. No entanto, devido ao tremendo sucesso que a interface HDMI obteve, de acordo com In-Stat referido em [6] foram vendidos 5 milhões de exemplares em 2004, 17.4 milhões em 2005, 63 milhões em 2006 e 143 milhões em 2007, tornou-se a interface standard para HDTV (High-Definition television), substituindo a interface DVI (Digital Visual Interface). Relativamente à interface DisplayPort, esta é utilizada em vários equipamentos, mas principalmente no sector dos computadores e vem complementar o HDMI. Contudo, comparando as duas interfaces previamente referidas, o HDMI tem algumas vantagens no que toca à capacidade de transmitir sinais CEC (Consumer Electronics Control) e a compatibilidade elétrica com o DVI. Mas o mais importante na realidade baseia-se na capacidade de transmissão dos sinais, sendo que o HDMI é capaz de fazer transmitir o sinal na sua largura de banda completa até 10 metros, enquanto que a DisplayPort apenas o consegue transmitir até 3 metros.

Através da implementação dos objetivos propostos pela dissertação será possível implementar um conversor HDMI capaz de fazer transmitir sinais de alto débito, tornando mais eficiente este tipo de comunicações e ao mesmo tempo fazendo-o de forma segura, protegendo as cópias e reproduções não autorizadas dos sinais transmitidos.

1.2 Motivação

Com a explosão que se fez sentir nos últimos anos na utilização do espetro de frequências, verifica-se que é necessário tornar a sua utilização mais eficiente no sentido de conseguir satisfazer a necessidade da sociedade de comunicar quase sem limites em termos de velocidade da comunicação em si. Promove-se assim uma nova abordagem do espetro de frequências, de maneira a que se possa utilizá-lo de uma forma mais eficaz. Ao longos dos anos tem-se vindo a verificar melhorias no que toca à eficiência espetral através do desenvolvimento e aplicação de algumas técnicas, tal como referido em [7], como por exemplo o QAM (Quadrature Amplitude Modulation) para modulação do sinal e também técnicas MIMO (Multiple Input Multiple Output) nas entradas e saídas do sistema de comunicação. Verificou-se que o aproveitamento do espetro de facto melhorou, no entanto, estas técnicas não são suficientes para se conseguir atingir um débito de algumas dezenas ou centena de Gb/s. Assim sendo, a solução passa por promover a utilização de bandas de frequência mais altas, contrariamente ao que se fez no passado.

Por definição, considera-se a banda de ondas mm entre 60 a 100 GHz e a banda THz entre 100 GHz a 1 THz. Estas bandas do espetro de frequências são bandas cuja utilização no passado foi

pouca ou até mesmo nenhuma, isto porque para conseguir explorar estas bandas são necessários componentes adequados à operação nas mesmas. Relativamente a banda de ondas mm, apesar de nos últimos anos terem sidos desenvolvidas e aplicadas técnicas que melhoram a eficiência espetral desta região, tal como referido anteriormente, a escassez da largura de banda limita o débito da ligação. Em [7] são referidas implementações realizadas no passado que conseguiram alcançar débitos até 100 GHz em ligações sem fios a uma distância de 1 metro com BER = 1 x 10-3, recorrendo também à utilização de mais de um transmissor e recetor. Apesar de inovadores estes valores revelam-se insuficientes para o que se pretende alcançar.

Quanto à região do espetro que corresponde a uma frequência superior a 10 THz, apesar da grande largura de banda disponível nesta região, existem várias limitações para a comunicação sem fios referidas em [8]. Destaca-se o facto do baixo balanço de potência possível para a transmissão devido aos limites de segurança dos olhos, os impactos atmosféricos na propagação do sinal (chuva, pó e poluição) e ainda o impacto da falta de alinhamento entre transmissores e recetores. Estas são algumas das razões que limitam a comunicação sem fios para frequências superiores a 10 THz.

Assim sendo, segundo [8], torna-se evidente que a banda do espetro com maior potencial para a comunicação sem fios é a banda entre 100 GHz e 1 THz, uma vez que não só oferece uma largura de banda bastante maior (desde GHz até alguns THz) comparativamente a outra bandas, mas também é uma região do espetro que não sofre muito devido às más condições atmosféricas. Para além disso, a utilização destas bandas de frequência altas acabará por aliviar o espetro relativamente à sua escassez e às suas limitações de capacidade.

Tendo em conta esta nova abordagem do espetro, o projeto iBrow tem vindo a desenvolver metodologias que permitem a manufaturação de transcetores para operar a estas frequências de baixo custo, mas que ao mesmo tempo são capazes de atingir altos débitos, para que desta maneira sejam integrados em redes de comunicação com e sem fios de grande velocidade. Os transcetores de baixo custo propostos pelo projeto passam por utilizar díodos ressonantes de efeito túnel (RTD) com formatos de modulação simples e com interligação com fibra ótica. Assim será possível satisfazer as necessidades previstas para 2020 de forma eficaz tanto em termos energéticos como financeiros.

Para que se possa demonstrar o potencial desta tecnologia proposta pelo iBrow, vai-se recorrer à transmissão de vídeo em alta definição descomprimido através destes mesmos dispositivos propostos pelo projeto. Assim sendo, para efetuar a transmissão será utilizada a interface HDMI, que fará transmitir um sinal de alto débito para de seguida o mesmo sinal ser transmitido pelos transcetores propostos pelo projeto iBrow. Esta transmissão terá de ser realizada em série visto que estes mesmos transcetores apenas suportam transmissão de dados em série, uma vez que esta é a maneira mais eficaz.

O HDMI é uma interface digital que transmite vídeo não comprimido e áudio que poderá ou não estar comprimido. Esta interface implementa vários protocolos entre quais se destaca o protocolo HDCP pois é o responsável pela prevenção de reproduções não autorizadas dos sinais a transmitir, o que é bastante importante hoje em dia dado os inúmeros consumidores que conse-

guem fazer cópias ilegais. Este protocolo faz uma verificação inicial antes de transmitir os dados encriptados no sentido de perceber se o dispositivo de destino é efetivamente um dispositivo autorizado para a reprodução de sinal. Esta é ainda uma interface que consegue transmitir sinais de alta definição e é ainda compatível com o DVI. Hoje em dia, esta é a interface standard para HDTVs e tem diversas aplicações tais como câmaras digitais, discos Blu-ray e leitores de DVD de alta definição, computadores pessoais, tablets e smartphones.

Em suma, esta implementação tornar-se bastante útil, uma vez que é capaz de abranger um vasto nível de aplicações, acessíveis a todos os utilizadores, tanto em ambientes empresariais como pessoais.

1.3 Objetivos

Este trabalho tem como principal objetivo a implementação de uma arquitetura de serialização e deserialização de um sinal HDMI, e que ao mesmo tempo faça o tratamento destes mesmos sinais para posterior envio e receção do sinal de alta velocidade em série. Como tal, será necessário utilizar um recurso que permita a implementação dessa mesma arquitetura versatilmente, por outras palavras, um recurso que permita eventuais reconfigurações da arquitetura desenvolvida e que ao mesmo tempo possua características que sejam úteis ao desenvolvimento do projeto.

O projeto fará uso então de uma FPGA VC7203 Virtex-7 que possibilita a implementação de uma arquitetura adequada e que ao mesmo tempo possui entradas e saídas de alta velocidade que vão ajudar na ligação do sinal com os transcetores de alta velocidade. O protótipo desenvolvido em hardware reprogramável deve ser devidamente validado para que o sinal digital possa de seguida ser transmitido através de uma ligação por fibra ótica usando os RTDs desenvolvidos no projeto iBrow.

1.4 Estrutura da Dissertação

FAZER ESTA PARTE !!!

Para além da introdução, esta dissertação contém mais x capítulos. No capítulo 2, é descrito o estado da arte e são apresentados trabalhos relacionados. No capítulo 3, ipsum dolor sit amet, consectetuer adipiscing elit. No capítulo 4 praesent sit amet sem. No capítulo 5 posuere, ante non tristique consectetuer, dui elit scelerisque augue, eu vehicula nibh nisi ac est.

Capítulo 2

Revisão Bibliográfica

Neste capítulo é realizada uma revisão bibliográfica das interfaces áudio e vídeo existentes, em específico do HDMI, também sobre métodos de codificação/descodificação de sinais HDMI numa FPGA e ainda sobre ligações de alta velocidade em série e cuidados que se deve ter com as mesmas.

2.1 Interfaces de transmissão de video/audio

As interfaces de áudio e vídeo definem parâmetros físicos e interpretações dos sinais recebidos, segundo [9]. Para sinais digitais a interface acaba por definir não só a camada física mas também a camada de ligação de dados e principalmente a camada da aplicação. As características físicas do equipamento (elétrico ou ótico) incluem o número e o tipo de ligações necessárias, tensões, frequências, intensidade ótica e ainda o design físico dos conectores. Relativamente à camada de ligação de dados, esta define como os dados da aplicação serão encapsulados para que, por exemplo, possam ser sincronizados ou para fazer correções de erros. Por fim, a camada da aplicação define o formato do sinal de áudio e vídeo a ser transmitido, normalmente incorporando *codecs* não específicos. No entanto, por vezes esta camada acaba por não definir em concreto o tipo de formato de dados deixando em aberto tal parâmetro para que se possa transmitir dados no geral (é o caso do HDMI). No caso dos sinais analógicos, todas as funções que existem para os sinais digitais definidas em três camadas, são representadas num único sinal.

No caso da transmissão de sinais de áudio e vídeo digital existem várias interfaces que passam a ser analisadas, segundo [9]:

- ***Display Port:*** utiliza um conector do tipo *DisplayPort* e é o principal concorrente do HDMI. Esta interface define uma interconexão sem licenças que foi inicialmente desenhada para ser utilizada numa conexão entre o computador e o monitor do mesmo. O sinal de vídeo não é compatível com DVI ou HDMI, mas um conector *DisplayPort* pode fazer passar estes sinais.
- ***IEEE 1394 "FireWire":*** utiliza um conector do tipo *FireWire* ou i.LINK. Este protocolo de transferência de dados é principalmente utilizado em câmaras digitais, mas também em

computadores e em transferências de sinal de áudio. Este tipo de interface é capaz de hospedar vários sinais no mesmo cabo entregando os dados nos devidos destinos.

- ***HDMI (High Definition Multimedia Interface):*** utiliza um conector do tipo HDMI e é uma interface de transmissão de sinal áudio/vídeo comprimida para transmissão de sinal digital descomprimida.

2.2 HDMI (*High Definition Multimedia Interface*)

O HDMI é uma interface de áudio e vídeo de alta definição que transporta dados áudio no formato não comprimido. Suporta num único cabo qualquer formato de vídeo em diversas resoluções e desde 2004 tem vindo a sofrer algumas alterações que vêm melhorar o desempenho da interface.

Esta interface está dividida em diversos canais de comunicação que implementam determinados protocolos, entre os quais se destacam as seguintes de [6]:

2.2.1 DDC - *Display Data Channel*

É um conjunto de protocolos utilizado nas comunicações digitais entre um dispositvo de origem e um dispositivo final que permite a comunicação entre ambos. Estes protocolos permitem que o ecrã comunique com o seu adaptador quais os modos que consegue suportar e também que o dispositivo que liga ao ecrã consiga ajustar alguns parâmetros, como por exemplo o contraste e a luminosidade. EDID (*Extended display identification data*) é a estrutura *standard* para este tipo de comunicações que define as capacidades do monitor e os modos gráficos suportados pelo mesmo. Este protocolo é utilizado pela *source* da comunicação do HDMI para obter os dados necessários do dispoistivo *sink*, no sentido de perceber quais os modos suportados pelo mesmo. Este canal é também ativamente usado para HDCP (*High-Bandwith Digital Content Protection*).

2.2.2 TMDS - *Transition-Minimized Differential Signaling*

É uma tecnologia utilizada para transmissão de dados em série de alta velocidade utilizado em comunicações digitais. O transmissor implementa um algoritmo que reduz as interferências eletromagnéticas nos cabos e permite ainda uma recuperação robusta de sinal de relógio no recetor.

Em específico na interface HDMI, este protocolo divide a informação a transmitir em 3 principais pacotes e intercala a sua transmissão: Período de transmissão de vídeo, período de transmissão de dados e período de controlo. No primeiro período (período de transmissão de vídeo) são transmitidas os pixeis do vídeo em linha. No segundo período (o período de transmissão de dados) são transmitidos os dados de vídeo e os dados auxiliares à transmissão dentro dos respectivos pacotes. O terceiro período ocorre entre os dois anteriores.

Para além de ser utilizada no HDMI, esta técnica é também utilizada em interfaces DVI.

2.2.3 CEC - *Consumer Electronics Control*

É uma característica do HDMI que permite ao utilizador controlar até 15 dispositivos que tenham esta mesma característica ativa e que estão conectados por HDMI usando apenas um controlo remoto. Também é possivel dispositivos inviduais controlarem outros dispositivos sem intervenção do utilizador

2.2.4 ARC - *Audio Return Channel*

Esta característica do HDMI utiliza 2 pins do conector. É uma ligação de audio que tem como objetivo substituir outros cabos entre a TV e outros recetores ou então sistema de som. Esta direção é usada quando é a TV que gera ou recebe o vídeo mas é outro equipamente que reproduz o som. Esta característica está apenas disponível a partir da versão 1.4 de HDMI.

2.2.5 HEC - *HDMI Ethernet Channel*

Esta especificação do HDMI, tal como a anterior, está também apenas disponível a partir da versão 1.4 do HDMI e é uma tecnologia capaz de consolidar vídeo, audio e dados em série num único cabo HDMI, permitindo também aplicações baseadas em IP sobre o HDMI e uma comunicação *Ethernet* bidireccional até 100 Mbit/s.

Uma das principais características mais recentes das interfaces HDMI prende-se ao facto de permitir que sinais não sejam reproduzidos em dipositivos não autrizados. Isto é, através de um protocolo cujo nome já foi referido anteriormente, HDCP (*High-Bandwith Digital Content Protection*), o sinal HDMI pode ser encriptado e posteriormente transmitido pela *source*, protegendo assim a sua reprodução em dispositivos não autorizados. Esta tem vindo a tornar-se uma característica importante, visto que a reprodução ilegal de vídeos tem vindo a tornar-se reccorrente nos dias atuais.

2.3 HDMI implementado sobre a FPGA

A interface HDMI, tal como descrito no subcapítulo anterior, consiste numa interface que permite a transferência de sinais áudio e vídeo digitais entre dois dispositivos, e como tal será necessário um adaptador que permita a conexão entre os dois dispositivos e que ao mesmo tempo sirva como recurso para a codificação e descodificação do sinal HDMI.

Assim sendo, existe *hardware* disponível que consegue fazer as duas funções descritas, nomeadamente em [2]. Esta interface HDMI consiste num adaptador e descodificador do sinal HDMI para a FPGA, sendo que são necessárias uma placa para a transmissão e outra para a receção do sinal. Cada placa tem respetivamente dois transmissores e dois recetores independentes e faz uso dos conectores FMC para se conectar com a FPGA.

2.3.1 Conexão à FPGA XILINX VC7203 Virtex-7

Na figura 2.1 da página 8 visualiza-se a placa de desenvolvimento a ser utilizada no projeto com numeração para as suas diversas características, sendo que nesta fase se pretende perceber o que são os conectores FMC (*FPGA Mezzanine Card*) e onde estão localizados nesta mesma placa.

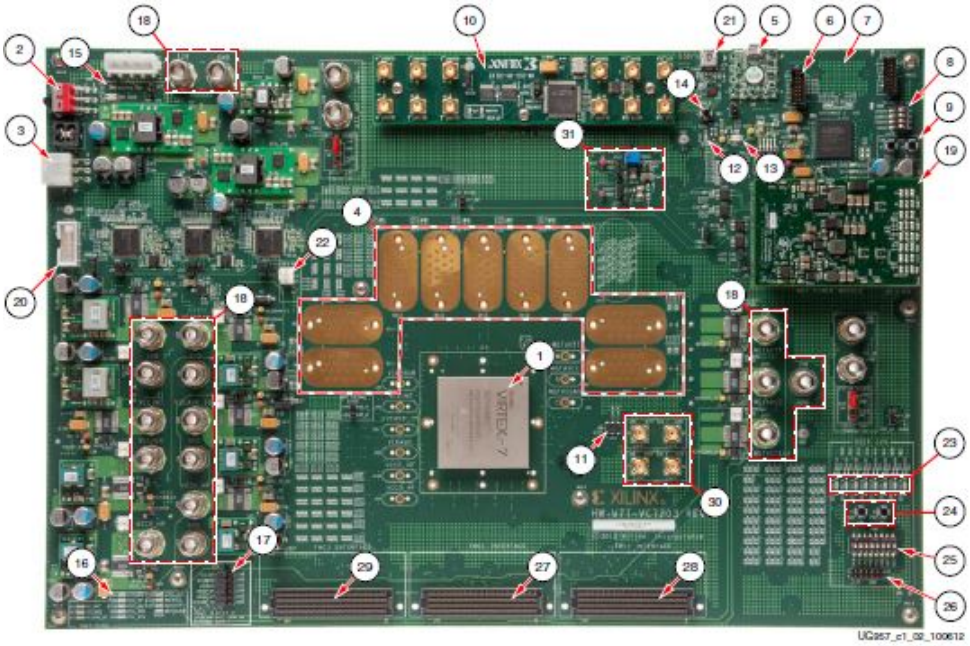

Figura 2.1: Vista Geral da FPGA VC7203 Virtex-7 retirada de [1]

A numeração 27, 28 e 29 correspondem aos conectores FMC disponíveis na FPGA a ser utilizada neste projeto. O número 27 corresponde ao conector JA2, 28 corresponde ao JA3 e 29 corresponde ao JA4. Estes conectores são usados como entradas e saídas de uma ligação, que neste caso em especifico será a uma placa HDMI, pois permitem uma ligação de alta velocidade (até 10 Gb/s). Estes três conectores HPC (High Pin Count) são compostos por 10x40 posições que permitem uma comunicação de alta velocidade por uma I/O cujo tamanho é relativamente pequeno.

O conector JA2 (FMC1 HPC) permite a seguinte conectividade:

- 68 pares que podem ser definidos pelo utilizador:
  - 34 pares LA
  - 17 pares HA
  - 17 pares HB
- 4 sinais de relógio diferenciais

O conector JA3 (FMC2 HPC) permite a seguinte conectividade:

- 68 pares que podem ser definidos pelo utilizador:

  - 34 pares LA
  - 17 pares HA
  - 17 pares HB
- 4 sinais de relógio diferenciais

O conector JA2 (FMC1 HPC) permite a seguinte conectividade:

- 65 pares que podem ser definidos pelo utilizador:
  - 34 pares LA
  - 16 pares HA
  - 15 pares HB
- 4 sinais de relógio diferenciais

Estes serão os conectores a ser utilizados e mais à frente neste relatório será explicado como é que os sinais são transmitidos.

2.3.2 Transmissor e Recetor

Este *hardware*, TB-FMCH-HDMI2, está dividido em 2 placas: o recetor (RX) que recebe o sinal recebido pelo cabo HDMI, faz a descodificação e envia o sinal para a FPGA, e o transmissor (TX) que faz o processo inverso, isto é, recebe o sinal proveniente da FPGA e transmite-o para o cabo HDMI para que possa chegar ao dispositivo de destino.

2.3.2.1 Recetor

Na figura 2.2 na página 10 é possível visualizar o diagrama de blocos do recetor disponível. As suas principais funções dividem-se nas seguintes:

1. **Receção do Sinal HDMI (ADV7612 para a FPGA localizada na placa)**

   A receção do sinal HDMI é feita por um conector HDMI e usa um circuito integrado ADV7612BSWZ-P que recebe sinal HDMI e retira do mesmo os sinais a serem passados para a FPGA localizada na placa HDMI. O recetor tem também uma memória EEPROM (electrically erasable programmable read-only memory) que é usada para guardar dados EDID.

2. **Interface com o conector FMC (da FPGA localizada na placa para o conector FMC)**

   Após passarem pela FPGA embebida (configurada por *default*) na placa são passados os seguintes sinais presentes na tabela 2.1 da página 10:

   Conclui-se que os dados presentes que são transmitidos para os conectores FMC, para além dos sinais de sincronização são essencialmente dados de vídeo, os do recetor 0 passados entre LA03_P a LA32_P e os do recetor 1 passados entre LA03_N a LA32_P. O sinail

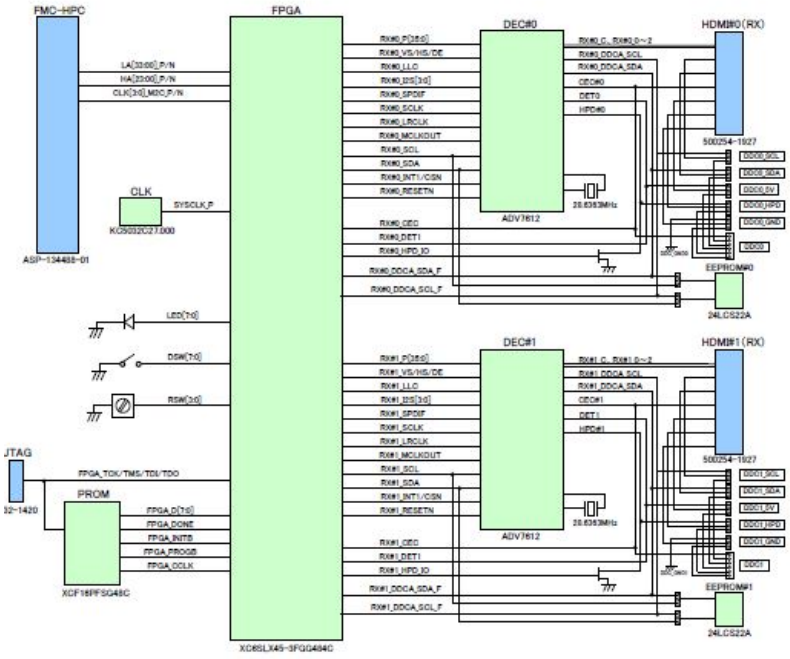

Figura 2.2: Diagrama de blocos de TB-FMCH-HDMI2 RX retirado de [2]

Nome do pin	***Input/0utput***	**FPGA para FMC**	**RX para a FPGA**
CLK0_M2C_P	*Output*	RX#0_LLC	RX#0 sinal LLC
CLK1_M2C_P	*Output*	RX#1_LLC	RX#1 sinal LLC
LA00_P_CC	*Output*	RX#0_VSYNC	RX#0_VSYNC
LA01_P_CC	*Output*	RX#0_HSYNC	RX#0_HSYNC
LA02_P	*Output*	RX#0_DE	RX#0 data enable
LA03_P a LA32_P	*Output*	RX#0_P0 a RX#0_P29	RX#0 dados de vídeo de 0 a 29
LA33_P	*Input/Output*	Não usado	————-
CLK0_M2C_N	*Input/Output*	Não usado	————-
CLK1_M2C_N	*Input/Output*	Não usado	————-
LA00_N_CC	*Output*	RX#1_VSYNC	RX#1_VSYNC
LA01_N_CC	*Output*	RX#1_HSYNC	RX#1_HSYNC
LA02_N	*Output*	RX#1_DE	RX#1 data enable
LA03_N a LA32_N	*Output*	RX#1_P0 a RX#1_P29	RX#1 dados de vídeo de 0 a 29
LA33_P	*Input/Output*	Não usado	————-
CLK2_M2C_P	*Input/Output*	Não usado	————-
CLK3_M2C_P	*Input/Output*	Não usado	————-
HA00_P a HA23_P	*Input/Output*	Não usado	————-
CLK2_M2C_N	*Input/Output*	Não usado	————-
CLK3_M2C_N	*Input/Output*	Não usado	————-
HA00_N a HA23_N	*Input/Output*	Não usado	————-

Tabela 2.1: Nomes dos pins da interface FMC de TB-FMCH-HDMI2 RX, adaptada de [2]

"*data enable*" é um sinal que sinaliza a chegada de dados. HSYNC é um sinal que representa a sincronização horizontal e é um pulso que sincroniza o início da linha do dispositivo de destino com a imagem que a originou. Por outro lado, o sinal VSYNC é a representação da sincronização horizontal, que faz o mesmo que HSYNC (mas na vertical), certificando-se de que o dispositivo de destino começa no topo na imagem na altura correta.

Uma nota importante ainda sobre a passagem dos sinais através dos conectores FMC é que os dados provenientes da FPGA embebida na placa para os conectores são amostrados na transição de 1 para 0 do sinal de relógio do vídeo, e como tal, estes mesmos dados devem ser lidos na transição de 0 para 1 do sinal do relógio do lado da FPGA principal. A figura 2.3 na página 11 ilustra esta situação.

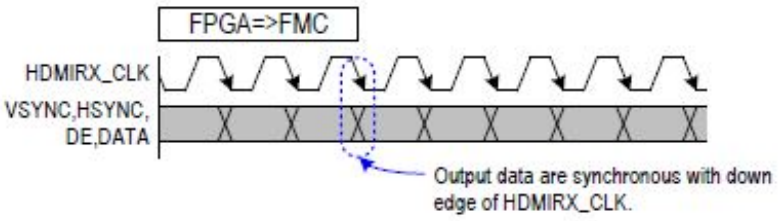

Figura 2.3: Amostragem dos dados provenientes da FPGA no recetor, retirada de [2]

Na figura 2.4 da página 11 é possivel visualizar a placa TB-FMCH-HDMI2 RX referida anteriomente.

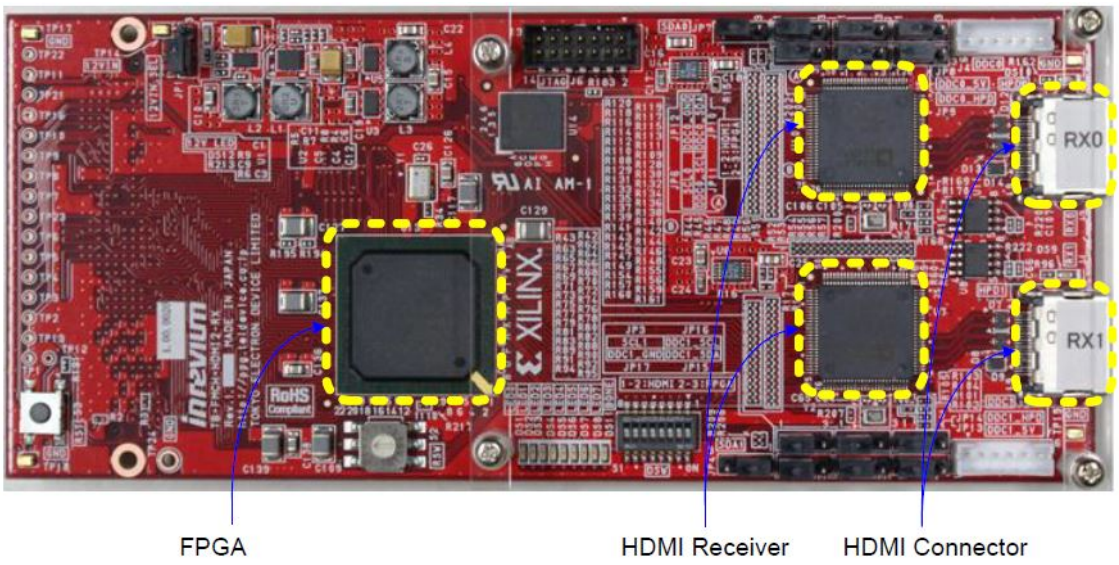

Figura 2.4: TB-FMCH-HDMI2 RX, retirada de [2]

2.3.2.2 Transmissor

O diagrama de blocos do transmissor está representado na figura 2.5 na página 12. Mais uma

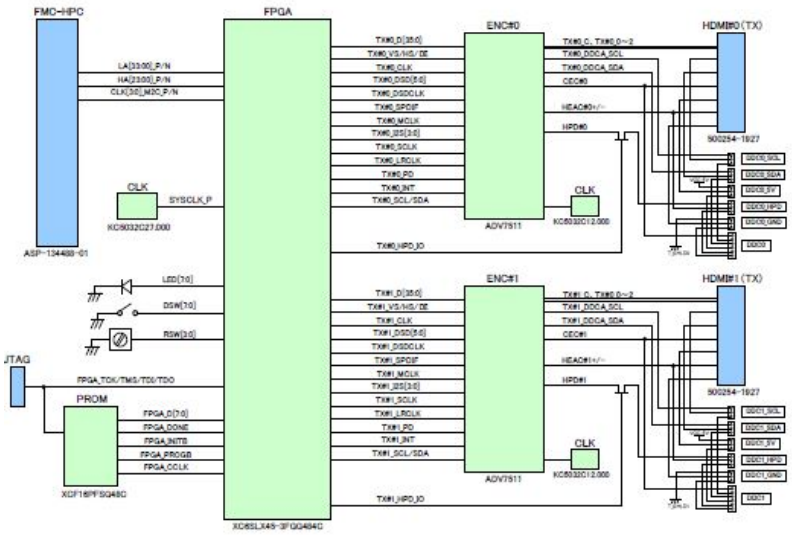

Figura 2.5: Diagrama de blocos de TB-FMCH-HDMI2 TX retirado de [2]

vez é possível dividir o diagrama em duas principais funções:

1. **Interface com o conector FMC (do conector FMC para a FPGA localizada na placa)**

   Os sinais representados na tabela 2.2 na página 13 são equivalentes aos sinais presentes na tabela 2.1 na página 10, mis uma vez com a placa configurada por *default*.

   Os dados capturados pela FPGA embebida na placa na transição de 0 para 1 do sinal de relógio do vídeo, tal como ilustra a figura 2.6 na página 12. Como tal, o sinal deve ser transferido na FPGA principal transição de 1 para 0 do sinal de relógio do vídeo.

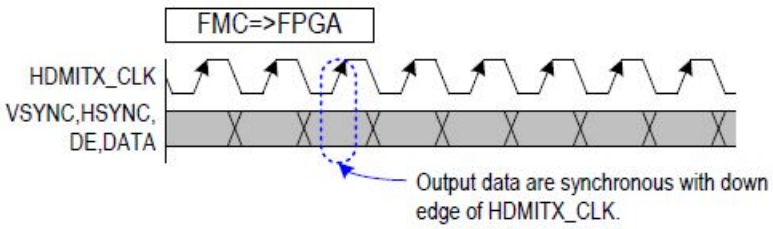

Figura 2.6: Amostragem dos dados provenientes do FMC no recetor, retirada de [2]

2. **Transmissor HDMI (da FPGA localizada na placa para ADV7511)**

   Após passar pela FPGA embebida na placa o sinal passa pelo bloco ADV7511 para de seguida ser possível o seu envio pelo conector HDMI para o dispositivo de destino.

Nome do pin	Input/0utput	FMC para FPGA	FPGA para TX
CLK0_M2C_P	Input	TX#0_DCLK	TX#0 sinal DCLK
CLK1_M2C_P	Input/Output	Não usado	———-
LA00_P_CC	Input	TX#0_VSYNC	TX#0_VSYNC
LA01_P_CC	Input	TX#0_HSYNC	TX#0_HSYNC
LA02_P	Input	TX#0_DE	TX#0 data enable
LA03_P a LA32_P	Input	TX#0_D0 a TX#0_D29	TX#0 dados de vídeo de 0 a 29
LA33_P	Input/Output	Não usado	———-
CLK0_M2C_N	Input	TX#1_DCLK	TX#0 sinal DCLK
CLK1_M2C_N	Input/Output	Não usado	———-
LA00_N_CC	Input	TX#1_VSYNC	TX#1_VSYNC
LA01_N_CC	Input	TX#1_HSYNC	TX#1_HSYNC
LA02_N	Output	TX#1_DE	TX#1 data enable
LA03_N a LA32_N	Output	TX#1_D0 a TX#1_D9	TX#1 dados de vídeo de 0 a 29
LA33_P	Input/Output	Não usado	———-
CLK2_M2C_P	Input/Output	Não usado	———-
CLK3_M2C_P	Input/Output	Não usado	———-
HA00_P a HA23_P	Input/Output	Não usado	———-
CLK2_M2C_N	Input/Output	Não usado	———-
CLK3_M2C_N	Input/Output	Não usado	———-
HA00_N a HA23_N	Input/Output	Não usado	———-

Tabela 2.2: Nomes dos pins da interface FMC de TB-FMCH-HDMI2 TX, adaptada de [2]

Na figura 2.7 da página 13 é possivel visualizar a placa TB-FMCH-HDMI2 RX referida anteriomente.

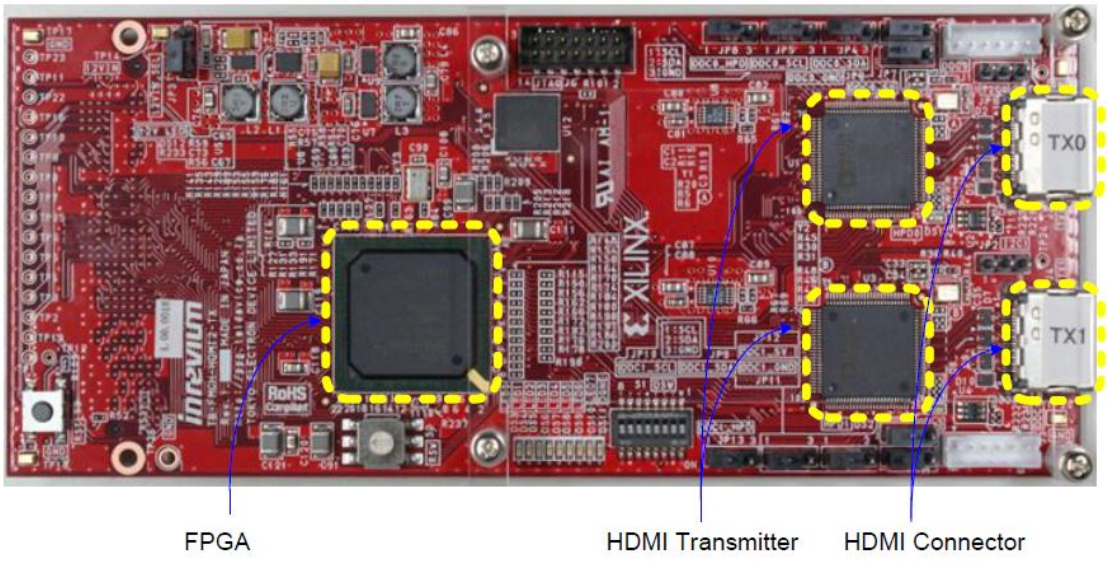

Figura 2.7: TB-FMCH-HDMI2 TX, retirada de [2]

2.3.2.3 Conexão DDC

Para esta configuração da placa, são suportadas dois tipos de conexão DDC que passam a ser descritas.

1. **Conexão Normal**

   Nesta configuração a comunicação DDC realiza-se normalmente entre o recetor e o transmissor HDMI, tal como ilustra a figura 2.8 na página 14. Existe um canal especifico para esta conexão que é recebido através do conector HDMI e enviado juntamente com os outros tipos de dados.

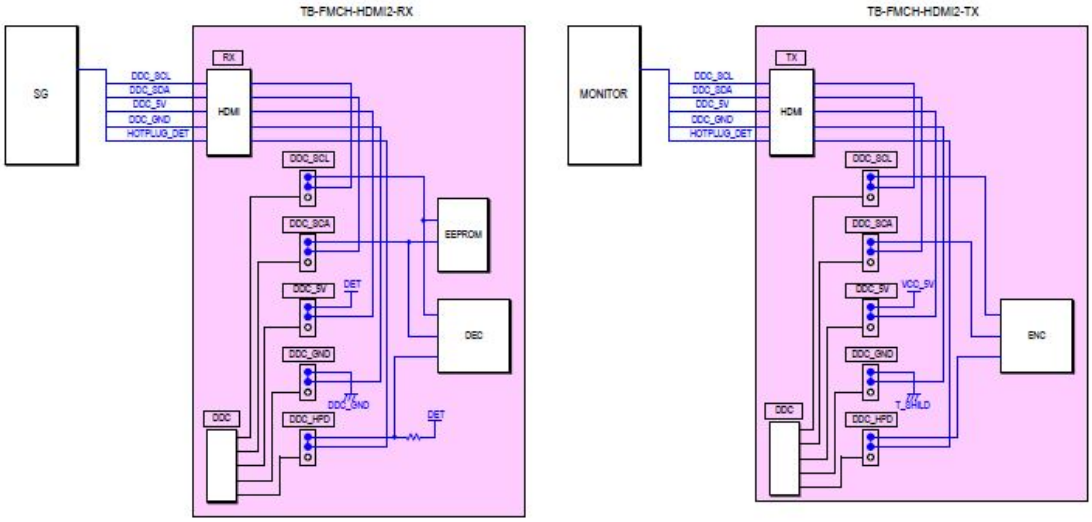

Figura 2.8: Configuração DDC normal, retirada de [2]

2. **Conexão "*through*"**

   Este tipo de conexão faz uma ligação direta do canal DDC entre o recetor e o transmissor inibindo a comunicação normal deste canal. Para fazer esta conexão é necessário um cabo DDC e fazer as conexões corretas. Esta situação está ilustrada na figura 2.9 na página 15.

   Este tipo de configuração poderá ser utilizada numa fase inicial do projeto a ser realizado, visto que não exige comunicação bidirecional através dos transcetores (onde os restantes sinais serão transmitidos), e como tal vem facilitar a comunicação entre dispositivo de origem e de destino.

2.4 Conexão de alta velocidade em série

A comunicação de dados de alta velocidade pode ser efetuada tanto em série como em paralelo, no entanto cada uma tem as suas implicações. No caso das comunicações em paralelo

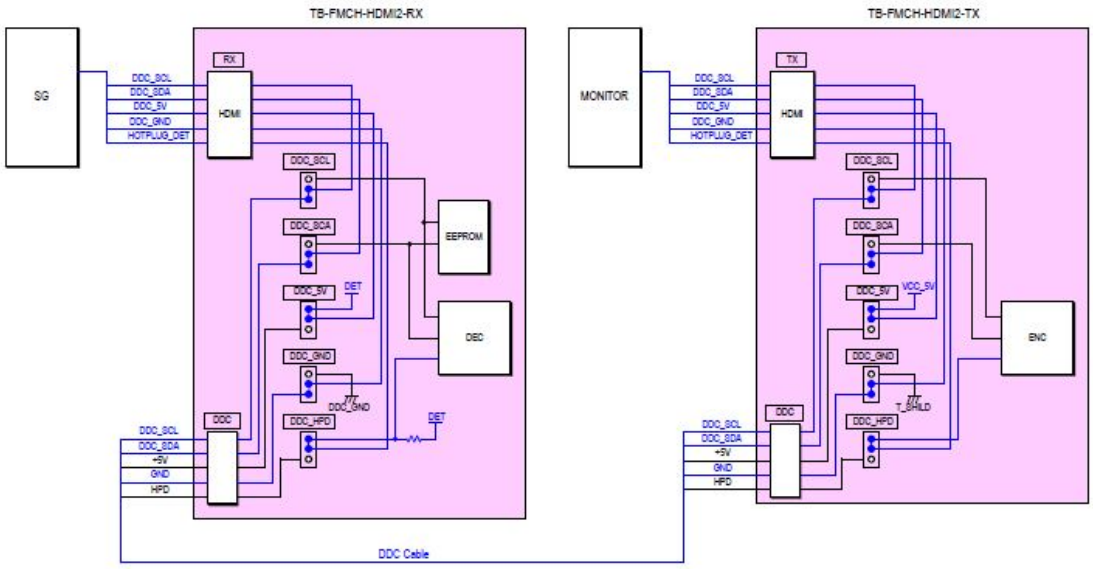

Figura 2.9: Configuração DDC "*through*", retirada de [2]

permitem uma velocidade de comunicação maior, em contrapartida tem um custo mais elevado devido à necessidade de mais recursos físicos e apresenta ainda problemas no que toca à diferença de tempos de chegada de dados e sinais de relógio (visto que estes podem chegar a tempos diferentes) e também no que toca à interferência entre canais.

Desta maneira, segundo [3], comunicações em série têm vindo a substituir as comunicações em paralelo em ligações de alta velocidade. Apesar disso, as comunicações realizadas dentro dos circuitos integrados são normalmente realizadas em paralelo, visto que permite maior rapidez de comunicação, e como tal é necessário a utilização de serializadores e deserializadores no sentido de transformar os dados nos diferentes domínios em que são utilizados.

2.4.1 Arquitetura de serializador e deserializador

Em [3] é apresentada uma arquitetura simples de um serializador/deserializador que passará a ser explicada em detalhe de seguida.

Na arquitetura representada no topo da figura 2.10 na página 16 está representado o serializador proposto em [3], cujas principais fases passam a ser descritas:

- Chegada do sinal em paralelo ao bloco "*data source*", que corresponde à chegada dos dados em paralelo a serem posteriormente transmitidos.

- Codificação da fonte (*source enconding*) é bloco que se segue nesta arquitetura e inclui construção de tramas, sincronização de padrões e ainda FEC [1].

[1] *Foward Error Correction* é uma técnica que permite o controlo de erros na transmissão de dados.

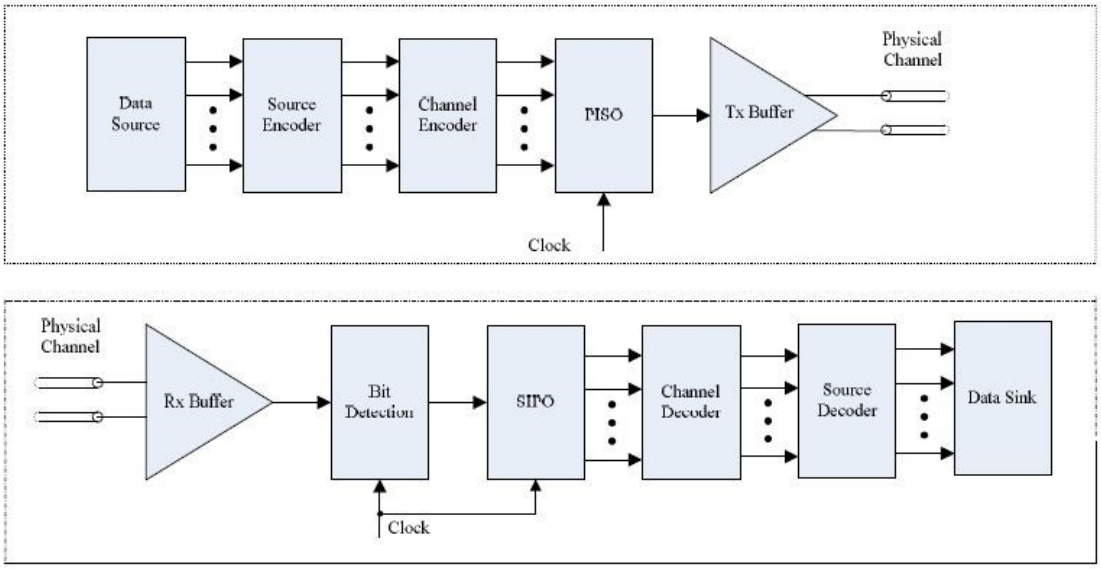

Figura 2.10: Arquitetura simples de um ser/des, retirada de [3]

- O bloco seguinte da arquitetura corresponde à codificação de canal, que é realizada de maneira a que o sinal a ser transmitido consiga um melhor desempenho no que toca a deteção de bits no recetor.
- De seguida, o sinal codificado é enviado para o bloco PISO (*parallel input - serial output)* e de onde sai um sinal em série dos dados a serem enviados.
- Finalmente estes dados são enviados para o *buffer* para que possam ser devidamente convertidos em sinais elétricos ou pulsos óticos para que de seguida sejam transmitidos pela camada física.
- Em alguns casos um equalizador (*pre-emphasis*) para corrigir alguns erros que possam acontecer no canal em ligações de alta velocidade é utilizado. Erros podem acontecer por diversos motivos, entre os quais: a dependência da atenuação e da dispersão relativamente à frequência, interferências e ruído.

O deserializador proposto segue a mesma estrutura que o serializador fazendo, no entanto, o processo inverso.

2.4.2 Considerações na implementação deste tipo de arquitetura

Em [3] são apontados os principais desafios na implementação deste tipo de arquitetura que passarão a ser descritos brevemente e que serão tidos em conta.

1. **Restrições na utilização de circuitos**

Logo à partida existem grandes restrições no que toca aos circuitos utilizados nestes tipos de serializadores e deserializadores. Isto porque os sinais recebidos em paralelo são sinais digitais, contudo, quando estes sinais passam pelo canal de transmissão sofrem distorções e também lhes é adicionado ruído, o que leva a que o sinal recebido do lado do recetor seja um sinal analógico e que necessite de ser tratado como tal. A sua recuperação tem de ser então baseada na regeneração correta do sinal de relógio e também na amostragem apropriada.

Ao mesmo tempo, este tipo de dispositivos são normalmente um subsistema de um sistema grande e usados em dispositivos portáteis, e como tal precisam de ter um baixo consumo de energia. Assim sendo, um dos primeiros grandes desafios desta implementação de serializador/deserializador, segundo [3], é conseguir implementar circuitos digitais de alta velocidade e que ao mesmo tempo têm um baixo consumo de potência. Esta mesma referência apresenta duas principais técnicas utilizadas para alcançar tais objetivos que passam pela utilização de lógica CMOS que permitem a utilização a alta velocidade com baixo consumo de potência.

Outro requisito crítico na implementação deste tipo de arquiteturas é também a adaptação das impedâncias características do *buffer* (de transmissão e receção) com a impedância característica da linha onde é transmitido o sinal. Isto porque, caso estas não estejam adaptadas ocorrerão reflexões no lado do transmissor ou do recetor (consoante a desadaptação) que não permitem a transmissão da potência total do sinal. No entanto, este requisito requer um grande consumo de potência, pois na prática os canais de transmissão têm uma impedância muito baixa.

2. **PISO (*Parallel input – serial output*) e SIPO (*serial input – parallel output*)**

   Estes blocos são necessários para o correto funcionamento deste tipo de arquitetura uma vez que é responsável pela conversão dos dados que se fazem chegar em paralelo em série e vice-versa. Na figura 2.11 na página 18 apresenta-se algumas topologias de PISO e SIPO apresentadas em [3]:

   No circuito a) visualiza-se uma estrutura de um único andar que é demasiado lenta devido às capacidades intrínsecas largas no nó de conversão. O circuito b) representa uma topologia heterogénea que é mais rápida que a anterior, e no circuito c) está representada uma topologia de árvore binária que é a topologia mais rápida das apresentadas.

   Para que estes blocos funcionem é necessário que exista um sinal de relógio de alta frequência (à taxa de débito do canal em série) e um sinal de baixa frequência também (para a os dados em paralelo). O sinal de relógio mais alto é usado para amostrar na saída os dados provenientes do sinal em paralelo e ao mesmo tempo para amostrar os dados recebidos em série. O sinal de frequência mais baixo, é utilizado para colocar na saída os dados que são amostrados do sinal em série. Deste modo, é necessário a utilização de multiplicadores de sinal de relógio e divisores de frequências que serão de seguida mencionados.

3. **Unidade de Multiplicação de Sinais de relógio (CMU – *Clock Multiplier Unit*)**

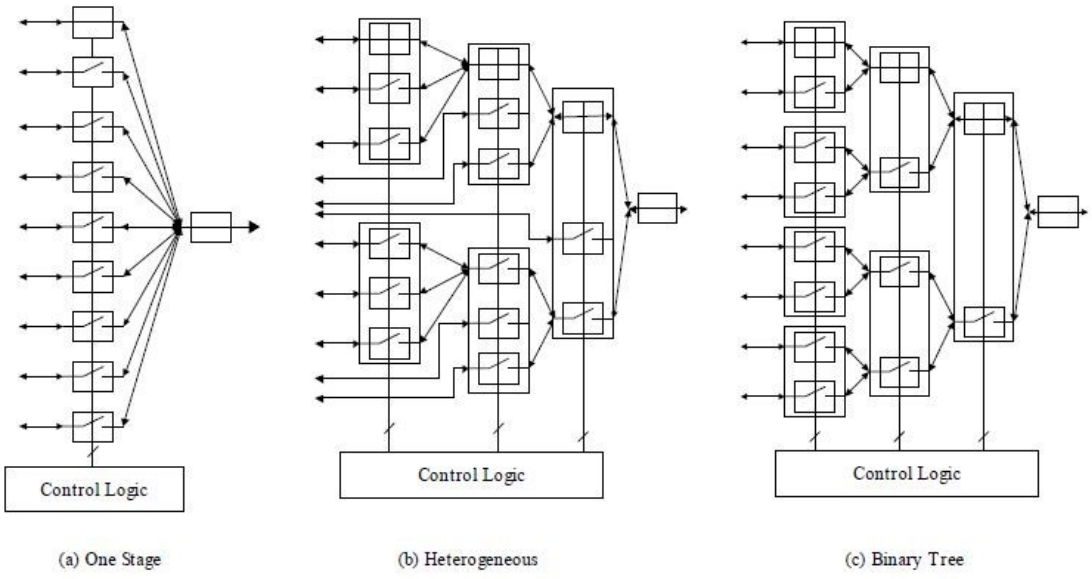

Figura 2.11: Arquitetura de PISO/SIPO, retirada de [3]

O sinal de relógio de alta frequência é bastante importante na implementação de arquiteturas de serialização e deserialização de alta velocidade, isto porque este sinal é necessário tanto do lado do recetor como do transmissor, segundo a fonte [3]. Do lado do transmissor é necessário para gerar os símbolos a serem transmitidos e do lado do recetor é necessário para que a amostragem do sinal recebido possa ser bem realizada. É comum que este sinal de relógio seja então partilhado entre o recetor e o transmissor, sendo que será necessário um bloco que faça o ajuste de fase deste sinal do lado do recetor. Este é necessário por causa do atraso introduzido pelo canal, que não é conhecido à priori e também devido ao ruído introduzido no canal que tornam a fase do sinal recebido bastante crítica para o desempenho do transcetor. Esta unidade é então responsável por tal procedimento.

4. **Equalização**

O sinal comunicado ao longo do canal pode sofrer interferências por vários motivos, interferências essas que são críticas no que toca à receção do sinal no recetor. Como tal, existe uma necessidade de utilizar técnicas que melhorem a ligação entre os dois terminais. Segundo [3], esta melhoria pode ser realizada através da utilização de canais de ligação de melhor qualidade. No entanto, esta opção traz custos acrescidos à ligação.

Por outro lado, também pode ser utilizada uma técnica de equalização que consegue obter bons resultados sem custos acrescidos à ligação. Ainda na mesma referência são apresentadas diferentes combinações de métodos de equalização que passam a ser brevemente descritos:

- Linear ou não-linear

- Pode ser implementado do lado transmissor, ou do recetor ou ainda de ambos os lados
- Pode ser implementado em tempo contínuo ou discreto
- Pode ser adaptativo ou fixo

Assim sendo, existe um vasto conjunto de opções de equalização que estão diretamente relacionadas com o circuito CDR (*Clock Data Recovery*), sendo que as mais importantes serão referidas mais à frente neste relatório.

5. **CDR (*Clock Data Recovery*)**

   Tal como referido anteriormente, a comunicação de sinais de alta velocidade pode sofrer diversas interferências durante a sua transmissão. Contudo, segundo [3], após a equalização do sinal estas mesmas interferências são parcialmente compensadas permitindo assim uma recuperação dos dados transmitidos. Para fazer a correta recuperação do sinal é necessário recorrer a um circuito que recupere inicialmente o sinal de relógio transmitido do emissor para que o sinal recuperado possa ser usado para recuperação dos dados transmitidos. Uma estrutura deste tipo de circuito será descrita mais à frente neste relatório.

2.4.3 Serializador e Deserializador disponíveis na FPGA

As FPGA de série 7 da XILINX têm disponíveis transcetores capazes de comunicação em série de alta velocidade, tal como é necessário neste projeto. Em específico, na FPGA XILINX VC7203 Virtex-7 estão disponíveis transcetores GTX que permitem uma velocidade de 10 Gb/s e que são os mais adequados para conexão à fibra ótica. Noutros modelos existem outros transcetores, como por exemplo GTZ (que permite até 28 Gb/s), GTH (que permite débitos até 13,1 Gb/s) e GTP (com débitos até 6,6 Gb/s). No entanto apenas serão abordados os transcetores GTX, visto que são os mais adequados para este tipo de comunicações.

Na figura 2.12 na página 20 é possível visualizar a FPGA a ser utilizada no projeto e visualiza-se ainda assinaladas as entradas GTX (QUAD_111, QUAD_112, QUAD_113, QUAD_114, QUAD_115, QUAD_116, QUAD_117, QUAD_118 e QUAD_119). Os transcetores baseiam-se na seguinte arquitetura, segundo [4]:

- **PMA (*Physical Medium Attachment Sublayer*)** que inclui:
  - suporte de taxas de débito até 12,5 Gb/s
  - uma PLL por canal para melhor flexibilidade do sinal de relógio
  - uma interface que faz a conversão de série para paralelo e de paralelo para série (PISO e SIPO)
  - uma PLL (*Phase-Locked Loop*)
  - Equalizador de decisão com feedback (DFE)
  - CDR (*clock data recovery*)

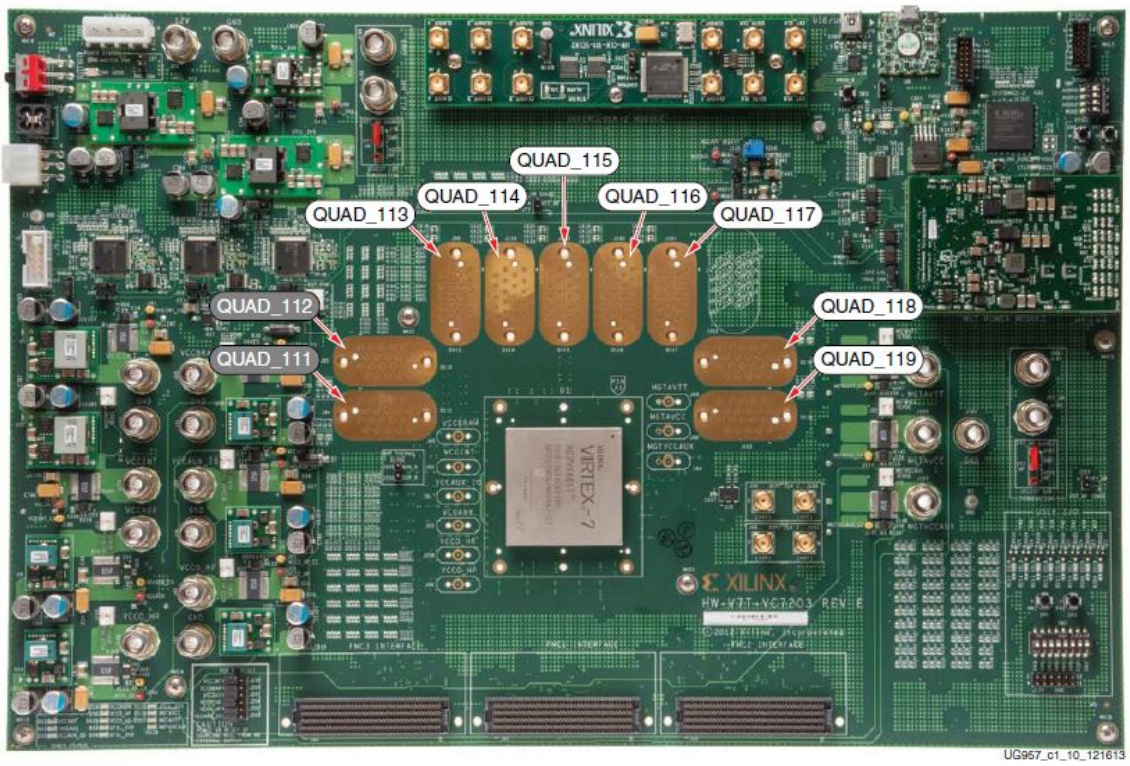

Figura 2.12: Identificação dos transcetores GTX na FPGA VC7203 Virtex-7, retirada de [1]

  - Bloco de pré-ênfase e equalização
  - Saída do transmissor programável

- **PCS (*Physical Coding Sublayer*) que inclui:**
  - *Datapath* de 2 e 4 byte internos para suportar diferentes taxas de débitos
  - Codificação e descodificação 8B/10B
  - Deteção de vírgula e alinhamento de palavra
  - PRBS (*Pseudo Random Bit Sequence*) gerador e verificador
  - FIFO para correção do sinal de relógio e ligação do canal
  - Lógica que processa os dados em paralelo reconfigurável
  - Este bloco trabalha com taxas de débitos de informação mais baixas.

É possível visualizar um diagrama geral da arquitetura dos transcetores GTX disponíveis na FPGA na figura 2.13 na página 21.

O transmissor e o recetor passam a ser descritos mais detalhadamente de seguida.

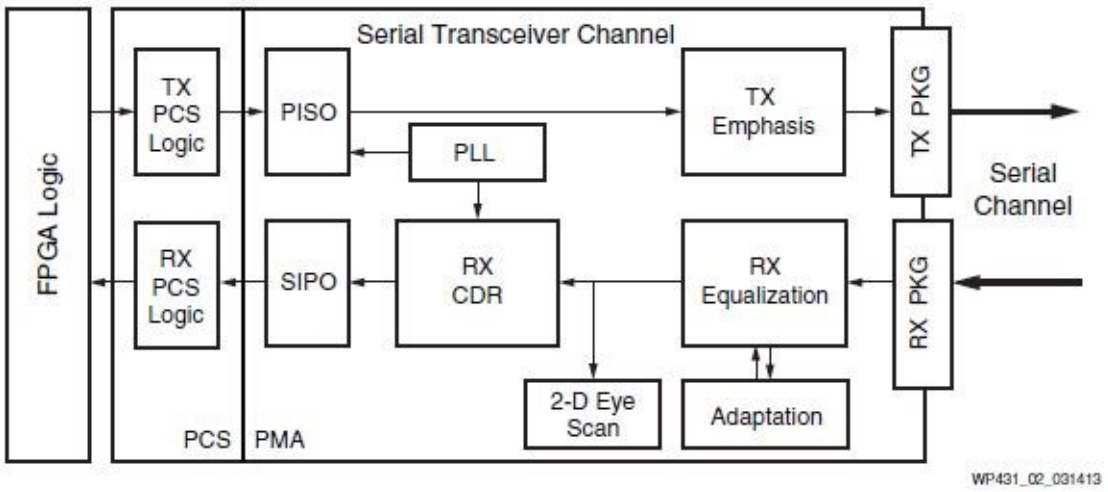

Figure 2: **7 Series Transceiver Channel Architecture**

Figura 2.13: Arquitetura geral dos transcetores GTX, retirada de [4]

2.4.3.1 Transmissor

Cada transcetor GTX inclui um transmissor independente que consiste num módulo PCS e um modulo PMA, tal como referido anteriormente. A figura 2.14 na página 21 representa o diagrama de blocos do transmissor.

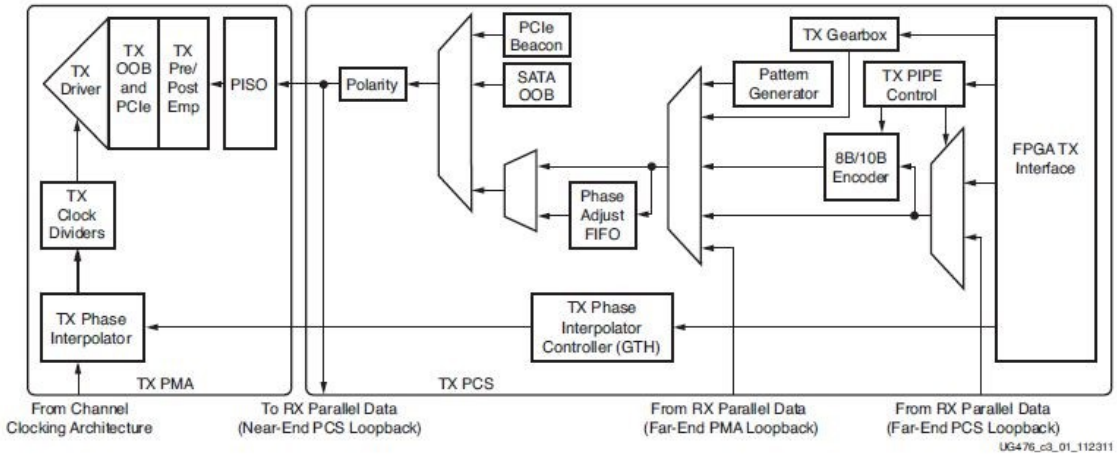

Figura 2.14: Diagrama de blocos de um transmissor GTX, retirada de [5]

Os dados provenientes da FPGA, cujo formato é em paralelo, passam para a interface transmissora, de seguida para os módulos PCS e PMA e, por fim, para a saída pelo driver do transmissor a alta velocidade.

O transmissor contém os seguintes blocos principais, cujas funcionalidades passam a ser brevemente resumidas, segundo [4]:

TX8B10BEN	TX_DATA_WIDTH	TX_INT_DATAWIDTH	Tamanho na interface da FPGA (bits)	Tamanho interno dos dados (bits)
1	20	0	16	20
	40	0	32	20
	40	1	32	40
	80	1	64	40
0	16	0	16	16
	20	0	20	20
	32	0	32	16
	32	1	32	32
	40	0	40	20
	40	1	40	40
	64	1	64	32
	80	1	80	40

Tabela 2.3: Configuração do tamanho dos dados de TXDATA, adaptada de [5]

1. **Interface transmissora da FPGA**

   Esta interface serve como porta de comunicação entre a FPGA e o datapath do transmissor. Esta comunicação é feita através da escrita de dados na porta TXDATA na transição de 0 para 1 do sinal de relógio TXUSRCLK2.

   O tamanho do sinal a ser transmitido pode ser configurado para 2, 4 ou 8 bytes. Na realidade este tamanho é definido por TX_DATA_WIDTH e TX_INT_DATAWIDTH e ainda por TX8B19BEN, e o tamanho interno destes sinais pode ser 16, 20, 32, 40, 64 e 80 bits. A tabela 2.3 na página 22 demonstra como esses tamanhos são configurados através das entradas referidas.

   Quando o codificador 8B/10B está ativo, então TX_DATA_WIDTH deve estar configurado para 20, 40 ou 80 bits e nesta situação, a interface do transmissor com a FPGA apenas utiliza os dados provenientes da porta TX_DATA_WIDTH. Quando o mesmo está desativo, então TX_DATA_WIDTH pode estar configurado para 16,20,32,40,64 ou 80 bits.

   O sinal TX_INT_DATAWITH é um atributo que configura a ativação do datapath de 2 e 4 bytes disponível internamente no transcetor.

   Para além do sinal de relógio TXUSRCLK2, que é o sinal de relógio principal para a sincronização dos sinais que chegam ao transmissor, existe um segundo sinal de relógio paralelo que é usado internamente para operações lógicas a realizar no módulo PCS. Este segundo sinal de relógio, TXUSRCLK, irá depender do tamanho interno dos dados usado no datapath e da taxa de transmissão do transmissor GTX. É possível calcular esta mesma taxa através da divisão entre a taxa de transmissão da linha e do tamanho interno dos dados utilizado no datapath. Para além disso, estes dois sinais de relógio têm uma relação fixa que determina os seus valores que depende dos valores presentes em TX_DATA_WIDTH e TX_INT_DATAWIDTH. Essas relações estão apresentadas na tabela 2.4 na página 23:

Tamanho na interface da FPGA (byte)	TX_DATA_WIDTH	TX_INT_DATAWIDTH	Frequência de TXUSRCLK2
2	16, 20	0	f(TXUSRCLK2) = f(TXUSRCLK)
4	32, 40	0	f(TXUSRCLK2) = f(TXUSRCLK) / 2
4	32, 40	1	f(TXUSRCLK2) = f(TXUSRCLK)
8	64, 80	1	f(TXUSRCLK2) = f(TXUSRCLK) / 2

Tabela 2.4: Configuração da frequência de TXUSRCLK2, adaptada de [5]

Assim sendo, é possível fazer uso dos transcetores disponíveis na FPGA, utilizando um tamanho adequado de dados para a transmissão, tendo em conta as configurações necessárias e disponíveis para tal, tal como descrito anteriormente. Por outras palavras, utilizando tamanhos de entradas devidamente apropriados, será fácil enviar os dados recebidos e descodificados do HDMI através destes transcetores, que pode vir a ser útil numa fase inicial do projeto.

2. **Codificador 8B/10B do transmissor**

   Esta é a codificação utilizada no sinal para de seguida fazê-lo enviar pelas portas de alta velocidade, e é uma codificação *standard* que troca dois bits por byte para alcançar um equilíbrio e obter uma disparidade limitada para que a recuperação de relógio seja razoável. O transmissor possui um *datapath* específico para fazer este tipo de codificação e ao mesmo tempo poupar recursos da FPGA apesar de aumentar a latência no caminho do transmissor.

   A ativação ou não deste bloco é representada no sinal TX8B10BEN. Quando está ativo, o sinal passado na interface do transmissor com a FPGA por TXDATA é codificado antes de ser enviado pelas saídas de alta velocidade, caso contrário, tal não acontece e o sinal é enviado tal como é transmitido.

3. ***Gearbox* do transmissor**

   Este bloco suporta a codificação do sinal 64B/66B e 64B/67B, uma vez que este tipo de codificação é utilizada em alguns protocolos de comunicação de alta velocidade. Esta utilização permite reduzir a sobrecarga da codificação 8B/10B e ao mesmo tempo reter os benefícios de um esquema de codificação. Este bloco suporta interfaces de 2, 4 ou 8 bytes e a codificação dos dados é feita na lógica da FPGA.

4. ***Buffer* do transmissor**

   O transmissor GTX tem disponível também um buffer e um bloco de alinhamento de fase no seu circuito para que possa sincronizar os diferentes domínios dos sinais de relógio. Isto acontece porque internamente o transmissor tem dois sinais de relógios paralelos: o sinal do domínio PMA (XCLK) e o sinal de relógio TXUSRCLK. No entanto, quando a transmissão de dados entre estes dois domínios é realizada é necessário que os sinais estejam

sincronizados e as diferenças de fase resolvidas. Como tal, será necessário a utilização deste bloco para se poder realizar a sincronização entre os sinais.

O circuito de alinhamento da fase é utilizado para resolver a diferença entre as fases dos sinais quando o buffer não está ativo. Mas pelo menos um dos blocos deve ser sempre utilizado.

A utilização do *buffer* é mais fácil e é sempre recomendada a sua utilização, enquanto que o bloco de alinhamento de fase é um bloco mais complexo no que toca a lógica e que requer restrições adicionais nas fontes dos sinais de relógio. Por outro lado, o buffer não deve ser utilizado quando a latência é uma questão importante do circuito, uma vez que o bloco de alinhamento de fase consegue alcançar menor latência.

5. **Gerador de padrões do transmissor**

   A geração de sequência pseudoaleatórias é bastante utilizada em sistemas de telecomunicações para testar a integridade do sinal de ligações de grande velocidade. Apesar de estas mesmas sequências parecerem aleatórias à primeira vista, na realidade apresentam determinadas características que são utilizadas para medir a qualidade da ligação. Este bloco do transmissor é responsável por esta ação.

6. **Controlo de polaridade do transmissor**

   Este bloco é responsável por inverter os dados em paralelo antes da sua serialização e transmissão para compensar a inversão de polaridade no par diferencial, isto porque tal pode levar a uma inversão de polarização dos dados transmitidos pelo GTX quando TXN e TXP são acidentalmente trocados na PCB.

7. **Controlo do sinal de relógio de saída do transmissor**

   Este bloco é responsável pelo controlo da divisão do sinal de relógio em série e pelo controlo da divisão e seleção do sinal de relógio em paralelo.

   No que toca à divisão do sinal de relógio em série, cada módulo PMA do transmissor possui um divisor D que divide o sinal de relógio da PLL para suportar taxas de transmissão mais baixas. Este divisor pode ser definido estaticamente para aplicações com uma taxa de transmissão fixa, mas também pode ser utilizado para ligações onde a taxa de transmissão pode variar e como tal D varia com essa mesma variação. Este bloco é responsável por esta divisão do sinal de relógio em série.

   Quanto à divisão e seleção do sinal de relógio paralelo, o sinal de relógio em paralelo que sai deste bloco de controlo de divisão de relógio pode ser utlizado como o relógio de fabrico lógico, dependendo das linhas de transmissão requisitadas. Este bloco controla também essa mesma divisão e seleção.

8. **Driver reconfigurável do transmissor**

É um *buffer* de saída diferencial de alta velocidade que possui características para maximizar a integridade do sinal, tal como controlo diferencial de tensão, pré-ênfase e resistências de terminação calibradas.

9. **Suporte de deteção de recetor para arquiteturas *PCI Express***

   As especificações das arquiteturas PCI Express incluem características que permitem detetar a presença do recetor para uma determinada ligação. Este bloco é responsável por esta mesma deteção.

2.4.3.2 Recetor

Cada transcetor possui um recetor independente, cujo diagrama de blocos está representado na figura 2.15 na página 25. Mais uma vez, o recetor possui dois módulos principais: PCS e PMA.

O sinal de alta velocidade em série chega ao RX ao modulo PMA, passa por PCS e, por fim, passa para a lógica da FPGA pela interface com a mesma.

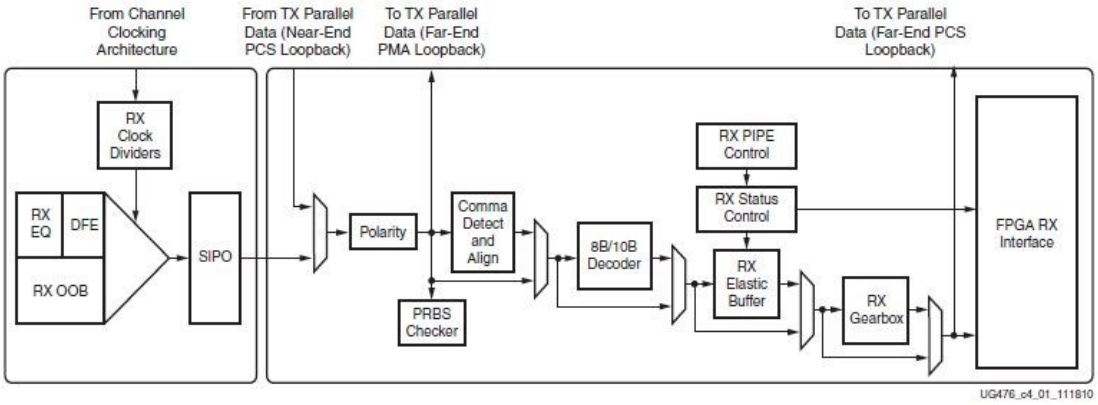

Figura 2.15: Diagrama de blocos de um recetor GTX, retirada de [5]

É possível dividir o recetor GTX nos seguintes principais blocos que passam brevemente a ser descritos:

1. ***Front End* Analógico do recetor**

   É um buffer diferencial de entrada de modo de corrente de alta velocidade que possui determinadas características tais como: reconfiguração de tensão de terminação do recetor e calibração das resistências de terminação.

2. **Equalizador do recetor**

   Ao longo da transmissão, diversos erros podem acontecer nos dados transmitidos e, como tal, são necessários filtros que permitam ou que pelo menos ajudem a realizar a recuperação do sinal recebido corretamente. Os transcetores GTX disponibilizam filtros adaptativos para tal recuperação: LPM que está otimizado para canais com poucas perdas e ainda DFE para

canais com perdas maiores. As arquiteturas apresentadas de seguida foram arquiteturas brevemente abordadas anteriormente dentro deste capítulo que estão referidas em [3].

Na figura 2.16 na página 26 encontra-se o diagrama de blocos do equalizador LPM. Este modo é recomendado para aplicações com débitos até 11,2 Gb/s de curto alcance, com perdas de canal até 12 dB à frequência de Nyquist.

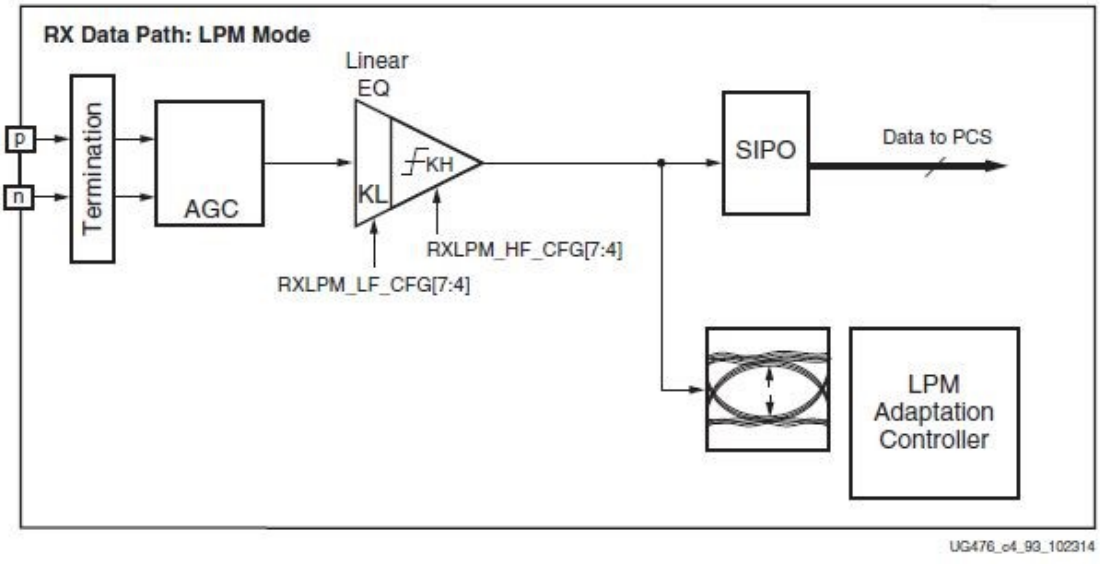

Figura 2.16: Equalizador em modo LPM, retirada de [5]

Na figura 2.17 na página 27 é possível visualizar o diagrama de blocos utilizado para o equalizador DFE (Decision Feedback Equalizer). Este é um filtro que utiliza a realimentação de símbolos detetados para produzir uma estimação da saída do canal. O DFE é alimentado com os símbolos já detetados e produz uma saída que é a combinação da saída do equalizador linear com estes mesmo símbolos já detetados.

Este equalizador é utilizado para ligações de média distância cujas perdas do canal rondam os 8 dB ou mais à frequência de Nyquist.

Vantagens da utilização deste tipo de equalizador:

- Efetua a equalização sem amplificação do ruído ou interferência
- Pode também fazer correções de reflexões causadas pelas descontinuidades do canal
- É vantajosa a sua utilização quando as interferências são preocupantes

Cuidados a ter na utilização deste tipo de equalizador:

- Este tipo de equalização deve ser cuidadosa quando não existe codificação de dados, uma vez que pode levar à não equalização ideal do sinal recebido (pois o filtro pode não se auto adaptar aos dados recebidos).

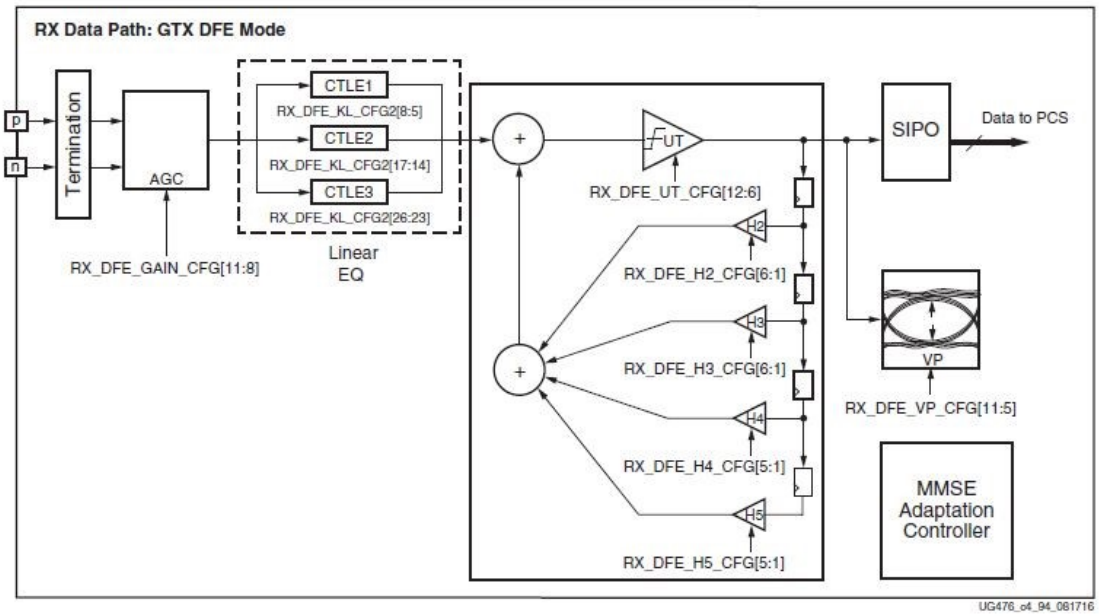

Figura 2.17: Equalizador em modo DFE, retirada de [5]

Visto que neste projeto se pretende realizar comunicações de média/longa distância, deve ser utilizado o equalizador DFE para obter uma boa equalização do sinal recebido.

3. **CDR (*Clock Data Recovery*) do recetor**

O circuito de CDR faz a recuperação do relógio dos dados recebidos em série. Na figura 2.18 na página 27 é possível encontrar a arquitetura deste mesmo circuito. Este circuito foi também já brevemente referido no subcapítulo anterior que refere as considerações a tomar quando se implementam arquiteturas de serialização e deserialização, e passa de seguida a ser descrito.

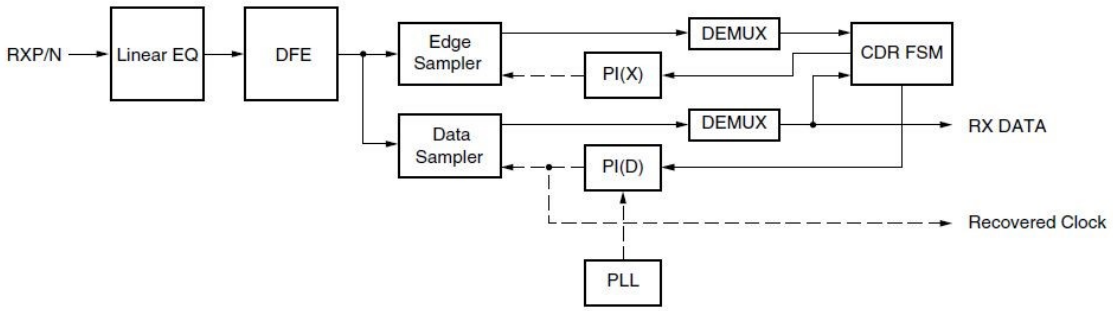

Figura 2.18: Detalhes do circuito CDR (*Clock data recovery*), retirada de [5]

A tracejado encontra-se o caminho feito pelo sinal de relógio até à sua recuperação. Os

dados recebidos passam pelo equalizador e de seguida são capturados por um "data sampler" e um "edge sampler". O "edge sampler" captura a fase do sinal recebido em série quando este está na sua região de transição, enquanto que o "data sampler" captura a fase do mesmo sinal a meio do olho dos dados. Estas duas fases são de seguida enviadas para a máquina de estados do CDR para que esta consiga determinar a fase dos sinais que chegam e ao mesmo tempo controlar os interpoladores de fase (PIs).

4. **Controlo do sinal de relógio de saída**

   Tal como no transmissor, o bloco de divisão de sinal de relógio tem dois principais componentes: controlo da divisão do sinal de relógio em série e ainda controlo e seleção da divisão do sinal relógio em paralelo. As suas funções são iguais à do transmissor.

5. **Análise de Margem do Recetor**

   Com o aumento das taxas de débito da transmissão e também da atenuação os equalizadores dos recetores têm mais capacidade de superar a atenuação do canal. Contudo, isto traz um novo desafio, pois nestes casos, a qualidade da ligação não pode ser medida através da abertura do olho no diagrama de olho resultante.

   Para taxas de transmissão muito altas pode acontecer que o diagrama de olho do sinal recebido possa parecer completamente fechado, apesar de que após a equalização esteja aberto. Como tal, esta medida de qualificação da qualidade da ligação realizada pode então ter de ser reavaliada.

   Assim sendo, os transmissores GTX possuem um mecanismo que permite medir e visualizar a margem do diagrama de olho recebido após equalização. Também existem modos que permitem determinar e diagnosticar os efeitos das configurações de equalização.

   Este mecanismo permite que uma correta avaliação da qualidade do canal e para além disso, pode ser feita enquanto os dados estão a ser recebidos, devido ao seu mecanismo que assim o permite, não exigindo nenhuma alteração às configurações do recetor e nem nenhuma lógica extra da FPGA.

6. **Controlo da polarização do recetor**

   Tal como foi mencionado aquando a referência da funcionalidade deste mesmo bloco no transmissor, os sinais RXN e RXP podem ser trocados acidentalmente na PCB e como tal os dados diferencias recebidos pelo recetor estão invertidos. Este bloco é responsável pela inversão dos bytes em paralelo no módulo PCS antes da deserialização do sinal (SIPO) para compensar a polarização inversa do par diferencial.

7. **Verificador de Padrões do recetor**

   Este bloco é responsável pela verificação de determinados padrões PSBR e faz esta mesma verificação antes do alinhamento das palavras ou descodificação. Tal como descrito aquando a referência ao gerador destes mesmos padrões no transmissor, estes servem para verificar a integridade do sinal na ligação.

8. **Alinhamento de *Byte* e palavras do recetor**

Os dados em série que chegam ao recetor devem ser alinhados com limitações de símbolos antes de poderem ser utilizados como dados em paralelo. Assim sendo, o transmissor envia sequências reconhecíveis (normalmente são chamadas de vírgulas) e o recetor procura essa mesma sequência nos dados recebido. Quando a encontra move-a para os limites das palavras para que as palavras em paralelo recebidas sejam iguais às palavras enviadas pelo transmissor. Para ativar a utilização deste bloco o sinal de entrada RXCOMMADETEN deve ser verdadeiro, mas caso a latência seja um parâmetro critico do circuito então este bloco não deve estar ativo.

Para definir a sequência que o bloco de alinhamento deve procurar (a vírgula) nos dados que chegam em série ao recetor, então deve-se definir as entradas ALIGB_MCOMMA_VALUE, ALIGN_PCOMMA_VALUE e ALIGN_COMMA_ENABLE. Os tamanhos destas sequências dependerão dos valores em RX_DATA_WIDTH, que será explicado mais à frente neste relatório.

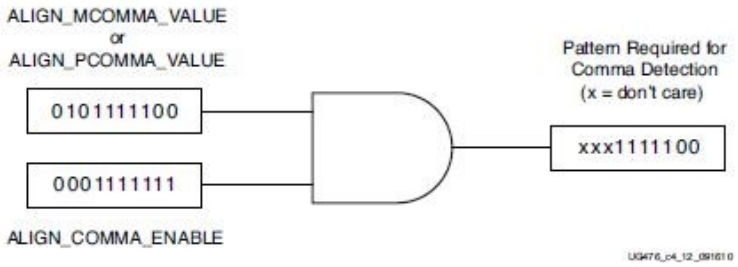

Figura 2.19: Mecanismo de obtenção da "vírgula", retirado de [5]

A figura 2.19 na página 29 ilustra o mecanismo utlizado para obter a sequência de procura dos dados recebidos no recetor quando ALIGN_COMMA_DOUBLE é falso. Quando este mesmo sinal é verdadeiro faz-se então uma extensão do sinal ALIGN_MCOMMA_VALUE e do sinal ALIGN_PCOMMA_VALUE, tal como está representado na figura 2.20 na página 30.

Quando este mesmo sinal está ativo os sinais de entrada ALIGN_MCOMMA_VALUE e ALIGN_PCOMMA_VALUE são combinados e o bloco procura por duas sequências de uma vez nos mesmos dados recebidos. Para ativar o alinhamento de palavras da sequência MCOMMA o sinal RXMCOMMAALIGNEN deve estar ativo, enquanto que para ativar o alinhamento da palavra PCOMMA o sinal RXPCOMMAALIGNEN deve estar ativo. Quando ambas estão ativas então o alinhamento é realizado com qualquer padrão.

É necessário ter em atenção que em aplicações cuja taxa de débito é superior a 5 Gb/s e que têm demasiado ruído, pode acontecer por vezes um falso alinhamento de palavras que leva à ativação do sinal. Isto indica que as palavras estão alinhadas sem realmente haver dados válidos presentes nelas. Assim sendo, neste tipo de sistemas é necessária a presença

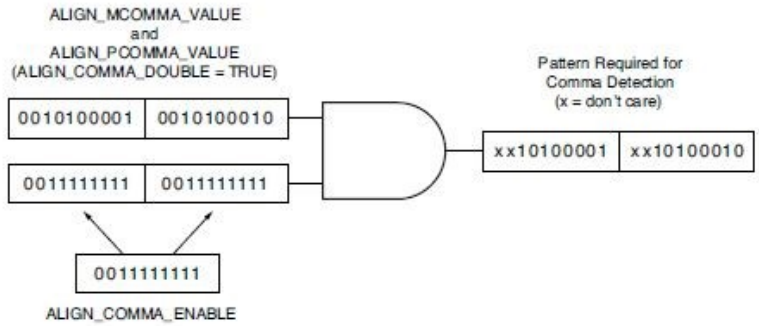

Figura 2.20: Mecanismo de obtenção da "vírgula" quando ALIGN_COMMA_DOUBLE=1, retirado de [5]

de um sistema que faça a verificação da validade destes dados alinhados para prevenir casos como estes.

Deste modo, com esta característica do recetor GTX da FPGA o alinhamento de palavras torna-se simplificado.

9. **Descodificador 8B/10B do recetor**

   Se os sinais estiverem codificados segundo 8B/10B então devem ser descodificados segundo esta norma também. Desta forma, o recetor possui um bloco responsável pela descodificação 8B/10B no recetor sem que gaste recursos adicionais à FPGA. Este mesmo bloco pode não estar ativo caso o sinal tenha codificação 8B/10B.

10. ***Buffer* do recetor**

    Tal como no transmissor o *buffer* é utilizado para possibilitar a sincronização entre o domínio do sinal de relógio do PMA em paralelo e o sinal de relógio RXUSRCLK. Isto porque, para ser possível a transmissão de dados entre os dois domínios a taxa do domínio PMA dever ser suficientemente parecida com a taxa de RXUSRCLK e todas as diferenças de fases entre as mesmas devem estar resolvidas. Este é, então, o bloco responsável por estes ajustes que devem ser feitos. Alternativamente a este buffer pode ser utilizado o circuito de alinhamento de fase, tal como foi referido anteriormente. No entanto, existem algumas vantagens e desvantagens de utilização destas duas opções.

    A utilização do *buffer* torna-se mais fácil em termos de operação, enquanto que o circuito de alinhamento exige lógica extra e restrições adicionais relativamente às fontes do sinal de relógio, tal como acontecia para o transmissor. Quanto à utilização de sinais de relógio o buffer pode usar tanto o sinal de relógio recuperado como o sinal de relógio local, enquanto que o circuito de alinhamento de fase apenas pode utilizar o sinal de relógio recuperado pelo recetor. Relativamente aos tempos de estabilização a utilização do buffer não necessita de começar a funcionar imediatamente após a sua inicialização, enquanto que o circuito de alinhamento do sinal necessita de esperar pela estabilização de todos os sinais de relógio

antes de conseguir realizar qualquer alinhamento de fase ou atraso. Em contrapartida, o buffer tem uma latência maior do que o circuito de alinhamento de fase, apesar de essa mesma latência depender também de algumas características do mesmo, como por exemplo a correção do sinal de relógio ou a ligação entre os canais do recetor.

11. **Correção do Sinal de relógio do recetor**

    Este bloco é responsável por evitar o overflow do buffer, isto porque o buffer faz a ponte de ligação entre dois domínios de sinal de relógio que apesar de serem muito idênticos nunca serão iguais. Como tal, haverá sempre uma ligeira diferença de fase entre os dois sinais causando acumulação que podem levar a um overflow ou underflow a não ser que seja corrigido.

    Para fazer esta correção, cada transmissor envia periodicamente um ou mais caracteres especiais que permitem que o recetor os elimine ou replique no buffer consoante a necessidade. Através da remoção desses caracteres quando o buffer está muito cheio e a sua replicação quando o *buffer* está vazio o recetor previne o *oveflow* ou *underflow*.

12. **Ligação de canais do recetor**

    Este bloco é responsável por fazer chegar todos os canais ao mesmo tempo ao recetor. Isto acontece porque existem protocolos que combinam múltiplos transcetores para criar um único canal de saída de alta velocidade. A diferença entre os tamanhos dos sinais de cada transcetor pode fazer com que os sinais sejam enviados todos ao mesmo tempo, mas que cheguem a tempos diferentes ao recetor. Este bloco é responsável por eliminar este efeito através do uso de um buffer como um bloco cuja latência é variável.

    Os transmissores GTX enviam um caracter que identifica a ligação entre canais (ou uma sequência de caracteres) simultaneamente. Quando este é recebido o recetor é capaz de determinar a diferença entre cada canal e ajustar a latência do buffer para que os dados cheguem todos ao mesmo tempo a interface com o utilizador.

13. ***Gearbox* do recetor**

    Este bloco é similar ao bloco gearbox do transmissor referido anteriormente neste relatório.

14. **Interface do recetor com FPGA**

    Este bloco é responsável pela comunicação entre o recetor e a FPGA, ou seja, a FPGA consegue ler os dados recebidos no recetor através da leitura do sinal RXDATA na transição de 0 para 1 do sinal de relógio RXUSRCLK2. O tamanho desta porta pode ser configurado para 2, 4 ou 8 bytes, mas a largura real da porta depende de RX_DATA_WIDTH, RX_DATAWIDTH e RX8B10BEN, tal como no transmissor. Assim, os tamanhos dos dados das portas pode ser 16, 20, 32, 40, 64 ou 80 bits.

    O recetor dos transcetores GTX contém datapaths internos de 2 e 4 bytes que são configuráveis através do atributo RX_INT_DATAWIDTH. A largura dos sinais na interface da

RX8B10BEN	RX_DATA_WIDTH	RX_INT_DATAWIDTH	Tamanho na interface da FPGA (bits)	Tamanho interno dos dados (bits)
1	20	0	16	20
	40	0	32	20
	40	1	32	40
	80	1	64	40
0	16	0	16	16
	20	0	20	20
	32	0	32	16
	32	1	32	32
	40	0	40	20
	40	1	40	40
	64	1	64	32
	80	1	80	40

Tabela 2.5: Configuração do tamanho dos dados de RXDATA, adaptada de [5]

FPGA é configurável através do sinal RX_DATA_WIDTH que deve ser 20, 40 ou 80 bits no caso de o descodificador 8B/10B estar ativo. Neste caso, a interface do recetor com a FPGA apenas usa as portas RXDATA. Quando o descodificador 8B/10B não é utilizado então RX_DATA_WIDTH pode ser configurado para outro qualquer valor disponível. O valor do tamanho dos sinais para as diferentes configurações no recetor está disponível na tabela2.5 na página 32.

A interface do recetor com a FPGA inclui dois sinais de relógio em paralelo: RXUSRCLK e RXUSRCLK2. A taxa de débito do sinal de relógio paralelo RXUSRCLK2 na interface é determinada pela taxa de débito de linha no recetor, a largura do sinal RXDATA e se a codificação 8B/10B está ativa ou não. Também um segundo sinal de relógio paralelo RXUSRCLK é disponibilizado para lógica interna no PCS do transmissor e depende da largura dos sinais internamente no datapath e da taxa de débito da linha do recetor. Esta taxa pode ser calculada através da razão entre a taxa de débito da linha e da largura dos dados internamente no *datapath*.

Existe uma relação entre o sinal de relógio RXUSRCLCK E RXUSRCLK2 fixa que se baseia nos sinais RX_DATA_WIDTH e RX_INT_DATAWIDTH. Por exemplo, para uma linha cuja taxa de débito seja superior a 6,6 Gb/s então é necessário recorrer ao datpath interno de 4 byte, ativando o sinal RX_INT_DATAWIDTH. A relação dos valores entre RXUSRCLK e RXUSRCLK2 está representada na tabela 2.6 na página 33.

Após a análise dos recursos existentes já na FPGA a ser utilizada neste trabalho concluiu-se que estes transcetores de alto débito permitem não só fazer a transmissão do sinal, mas também incluem técnicas de recuperação fiável dos mesmos dados transmitidos. Uma vez que estes transcetores são também bastantes flexíveis em termos de configurações tornará a

Tamanho na interface da FPGA (byte)	RX_DATA_WIDTH	RX_INT_DATAWIDTH	Frequência de TXUSRCLK2
2	16, 20	0	f(RXUSRCLK2) = f(RXUSRCLK)
4	32, 40	0	f(RXUSRCLK2) = f(RXUSRCLK) / 2
4	32, 40	1	f(RXUSRCLK2) = f(RXUSRCLK)
8	64, 80	1	f(RXUSRCLK2) = f(RXUSRCLK) / 2

Tabela 2.6: Configuração da frequência de TXUSRCLK2, adaptada de [5]

implementação da transmissão dos dados pretendidos mais fácil numa fácil inicial, antes da utilização dos transcetores desenvolvidos pelo projeto iBrow.

Capítulo 3

Capítulo Exemplo

Capítulo 4

Mais um Capítulo

Capítulo 5

Conclusões e Trabalho Futuro

Anexo A

Loren Ipsum

Bibliografia

[1] Xilinx and Inc, "VC7203 Virtex-7 FPGA GTX Transceiver Characterization Board User Guide (UG957)," 2014.

[2] Inrevium, *Manual do Utilizador de TB-FMCH-HDMI2 Hardware*. 2012.

[3] D. Chen, "SerDes Transceivers for High-speed Serial Communications,"

[4] Xilinx and Inc, "Xilinx WP431 Leveraging 7 Series FPGA Transceivers for High-Speed Serial I/O Connectivity, White Paper," 2013.

[5] Xilinx and Inc, "7 Series FPGAs GTX/GTH Transceivers User Guide (UG476),"

[6] Wikipedia Contributors, "HDMI," 2016.

[7] S. Koenig, D. Lopez-Diaz, J. Antes, F. Boes, R. Henneberger, A. Leuther, A. Tessmann, R. Schmogrow, D. Hillerkuss, R. Palmer, T. Zwick, C. Koos, W. Freude, O. Ambacher, J. Leuthold, and I. Kallfass, "Wireless sub-THz communication system with high data rate enabled by RF photonics and active MMIC technology," *2014 IEEE Photonics Conference, IPC 2014*, vol. 7, no. December 2013, pp. 414–415, 2014.

[8] J. Federici and L. Moeller, "Review of terahertz and subterahertz wireless communications," *Journal of Applied Physics*, vol. 107, no. 11, 2010.

[9] W. contributors, "audio and video interfaces and connectors," 2016.