Universidade Federal do ABC
Graduação em Engenharia de Informação

Paulo Koziol Candido da Silveira

ATUALIZAÇÃO DO SISTEMA DE TRANSMISSÃO DE DADOS DA SINALIZAÇÃO FERROVIÁRIA

para Companhia Paulista de Trens Metropolitanos em Rio Grande da Serra

Santo André - SP
2021

Paulo Koziol Candido da Silveira

ATUALIZAÇÃO DO SISTEMA DE TRANSMISSÃO DE DADOS DA SINALIZAÇÃO FERROVIÁRIA

para Companhia Paulista de Trens Metropolitanos em Rio Grande da Serra

Dissertação apresentada ao Curso de Graduação da Universidade Federal do ABC, como requisito parcial para obtenção do grau de Engenheiro de Informação.

Orientador: Profº Dr. Stilante Koch Manfrin

Santo André - SP
2021

Paulo Koziol Candido da Silveira

ATUALIZAÇÃO DO SISTEMA DE TRANSMISSÃO DE DADOS DA SINALIZAÇÃO FERROVIÁRIA

para Companhia Paulista de Trens Metropolitanos em Rio Grande da Serra

Dissertação apresentada ao Curso de Graduação da Universidade Federal do ABC, como requisito parcial para obtenção do grau de Engenheiro de Informação.

Aprovado em: ____/____/_____

BANCA EXAMINADORA

_____

Dr. Stilante Koch Manfrin, Professor Adjunto do Curso de Engenharia de Informação da UFABC

_____

Dr. Marco Aurélio Cazarotto Gomes, Professor Adjunto de Engenharia de Informação da UFABC

_____

Dr. Murilo Bellezoni Loiola, Professor Adjunto de Engenharia de Informação da UFABC

RESUMO

Este trabalho é um projeto de atualização da estrutura atual de comunicação do sistema de sinalização ferroviário da linha 10 da Companhia Paulista de Trens Metropolitanos na região da cidade de Rio Grande da Serra, este é responsável pela supervisão e controle do tráfego de trens, meio de transporte importante para a região metropolitana de São Paulo. Atualmente os dados são enviados por cabos metálicos sem nenhum tipo de multiplexação, sendo necessário uma quantidade grande de fios por longas distâncias, motivando ladrões a cometerem furto pelo preço da revenda e fácil comercialização do cobre no mercado ilegal. Com este projeto, os dados serão processados antes da transmissão criando uma codificação baseado no código de Hamming e multiplexados para envio serial em um canal óptico, aumentando a confiabilidade e diminuindo o custo do sistema, além de atenuar consideravelmente o interesse dos bandidos, já que a fibra óptica é constituída de vidro e não possui grande valor comercial. Por se tratar de um protótipo, serão construídos atuadores e sinalizadores para simular os equipamentos de via controlados e supervisionados pelo sistema.

Palavras-Chave: Comunicação Óptica. Sinalização Ferroviária. Fibra Óptica.

ABSTRACT

This work is a project to update the current communication structure of the railway signaling system of Line 10 of the Companhia Paulista de Trens Metropolitanos in the region of Rio Grande da Serra. It is responsible for the supervision and control of train traffic, important transportation to the metropolitan region of São Paulo. Nowadays data is sent by wire without any kind of multiplexing, requiring a large amount of wires over long distances, motivating thieves to steal for the resale price and easy commercialization of copper in the illegal market. With this project, data will be processed prior to transmission by creating Hamming code-based and multiplexing for serial sending in an optical channel, increasing reliability and lowering system cost, and considerably mitigating the interest of bandits, as optical fiber is It is made of glass and has no great commercial value. Because it is a prototype, actuators and beacons will be built to simulate the track equipment controlled and supervised by the system.

Keywords: Optical Communication. Railway signaling. Optical fiber.

Sumário

1. Introdução

1.1 História

Em 1867 foi inaugurada a primeira ferrovia paulista, chamada de São Paulo Railway. Impulsionada por capital privado tendo como liderança o Barão e Visconde de Mauá e parceria com a coroa Inglesa. Foi o principal meio de transporte para escoar toda a produção de café do interior, pois uma extremidade ficava na cidade de Jundiaí, até o porto de Santos, de onde eram carregados os navios para realizar a exportação [1][7].

Após o fim da concessão dos ingleses, em 1946, a linha passou a ser administrada pelo governo federal e em 1957 tornou-se subsidiária da Rede Ferroviária Federal SA (RFFSA) que também administrava outras ferrovias do país, recebendo o nome de Estrada de Ferro Santos-Jundiaí (EFSJ) [5].

Isso se deu até a década de 90, período de início do plano de desestatização do então presidente, Fernando Henrique Cardoso. O trecho de serra do mar foi concedido para a empresa de transporte de cargas MRS Logística e na região metropolitana o domínio ficou para a Companhia Paulista de Trens Metropolitanos (CPTM), subdividindo o trecho entre Luz até Jundiaí, chamado de Linha A e de Paranapiacaba até a Luz, nomeado de Linha D [6].

Hoje, a linha D passou a ser chamada de Linha 10 - Turquesa, transportando em média 340 mil passageiros por dia e liga a estação do Brás até Rio Grande da Serra [2].

Além da capital São Paulo atravessa importantes cidades da região do ABC Paulista, como São Caetano do Sul, Santo André, Mauá, Ribeirão Pires e por fim, a própria cidade de Rio Grande da Serra.

1.2 Justificativa

Dada a importância da Linha 10 da CPTM para a locomoção das pessoas que moram, trabalham ou estudam na região, qualquer problema que interfira no tráfego regular do sistema ferroviário, certamente causará impacto muito grande, além dos transtornos indiretos a todos os outros modais de transporte, já que os passageiros procurarão outras formas de locomoção [4].

Atualmente, o sistema de sinalização ferroviária da Linha 10 é responsável pelo controle e supervisão do tráfego de trens, permitindo que seja verificada a localização, indicando rota, mudanças de via e solicitando a parada das composições que trafegam na via.

Isso é possível devido aos sensores e atuadores, chamados de equipamentos de via, que se comunicam com pequenas salas de controle (bangalôs ou locações) ao longo do trecho por meio de sinais elétricos transmitidos em cabos de cobre.

Com essa mesma tecnologia de comunicação, há também a troca de dados entre bangalôs que chegam a ter quilômetros de distância entre eles, dependendo da região. Por esse motivo se faz necessária a utilização de várias bobinas de cabos, que são passados em tubulações subterrâneas protegidas por dutos de plástico rígidos.

Como estes cabos são feitos de cobre e que por sua vez, possuem considerável valor de revenda, isso é o suficiente para que haja diversas ocorrências de furtos. Oportunistas aproveitam as regiões menos habitadas e os horários em que não há circulação de trens para cavar, cortar e retirar os cabos (Figura 1).

Um ponto bastante crítico da Linha 10 é a região da estação de Rio Grande da Serra, a qual de janeiro a agosto de 2018, por exemplo, registrou 8 furtos de cabos no sistema de sinalização.

Figura 1: Registro de furto na região de Rio Grande da Serra (Arquivo Pessoal)

Na tabela 1 abaixo, constam a quantidade e os valores estimados que foram gastos para reposição dos cabos furtados somente no sistema de sinalização por ano nas 6 linhas administradas pela CPTM.

Tabela 1: Custo anual estimado para reposição de cabos furtados na sinalização (Fonte: base de dados da CPTM) [3]

Ano	Quantidade (m)	Custo para reposição
2015	2752,8	R$ 110.112,00
2016	1758,8	R$ 70.352,00
2017	3345	R$ 133.800,00
2018 (jan a mai)	1082,2	R$ 43.288,00

Os furtos dos cabos ocorrem em trechos relativamente pequenos, em torno de 100 à 200 metros, exigindo reparos com emendas, que mesmo sendo empregadas as técnicas corretas acabam aumentando a vulnerabilidade do cabo para entrada de água, diminuindo consideravelmente a sua vida útil.

Como dito anteriormente, os cabos ficam enterrados e em épocas de chuvas constantes as tubulações são inundadas, diminuindo a capacidade de isolação entre os fios, principalmente sobre os pontos de emendas, ocasionando falhas.

Dado todas estas problemáticas do sistema atual de comunicação por cabos metálicos, este trabalho tem como objetivo a substituição deste modelo por um sistema de comunicação óptica, que no decorrer deste relatório exibirá todas as etapas e estruturas necessárias para isto.

Outras empresas do ramo ferroviário, como a MRS Logística utilizam outras tecnologias para a transmissão de dados, que neste caso é o GPRS utilizando redes celulares para conseguir controlar da cidade de Juíz de Fora, em Minas Gerais, toda a malha ferroviária do país administrada pela companhia..

2. Objetivo

Assim, feito os esclarecimentos introdutórios pertinentes a esse estudo, este projeto tem por objetivo demonstrar a viabilidade de melhorias do sistema que hoje opera na Linha 10 - Turquesa da CPTM. Por causa do grande número de furtos no sistema de sinalização ferroviário na região de Rio Grande da Serra, a substituição do canal de comunicação de dados do tráfego ferroviário que hoje é feita por cabos de cobre por um cabo de fibra óptica.

Como as fibras ópticas são construídas a partir do vidro ou polímeros e possuem um valor de revenda menos atrativo do que o cabo metálico, a substituição destes por aqueles torna menos interessante a prática de furtos.

Outra vantagem da fibra óptica é a sua robustez à água que mesmo nas emendas subterrâneas, que são hermeticamente fechadas com uma estrutura plástica (Figura 2), é capaz de suportar essas condições sem maiores problemas, aumentando a confiabilidade do sistema.

Figura 2: Invólucro para armazenar emendas de fibra óptica (Fonte: Mercado Livre)

2.1 Análise da região

Na cidade de Rio Grande da Serra, onde se localiza o terminal da Linha 10, existem 2 bangalôs, chamados Rio Grande da Serra Norte e Rio Grande da Serra Sul, com aproximadamente 1700 metros de distância entre eles. No mapa da Figura 3 é possível visualizar o caminho dos dutos, indicados com a linha vermelha.

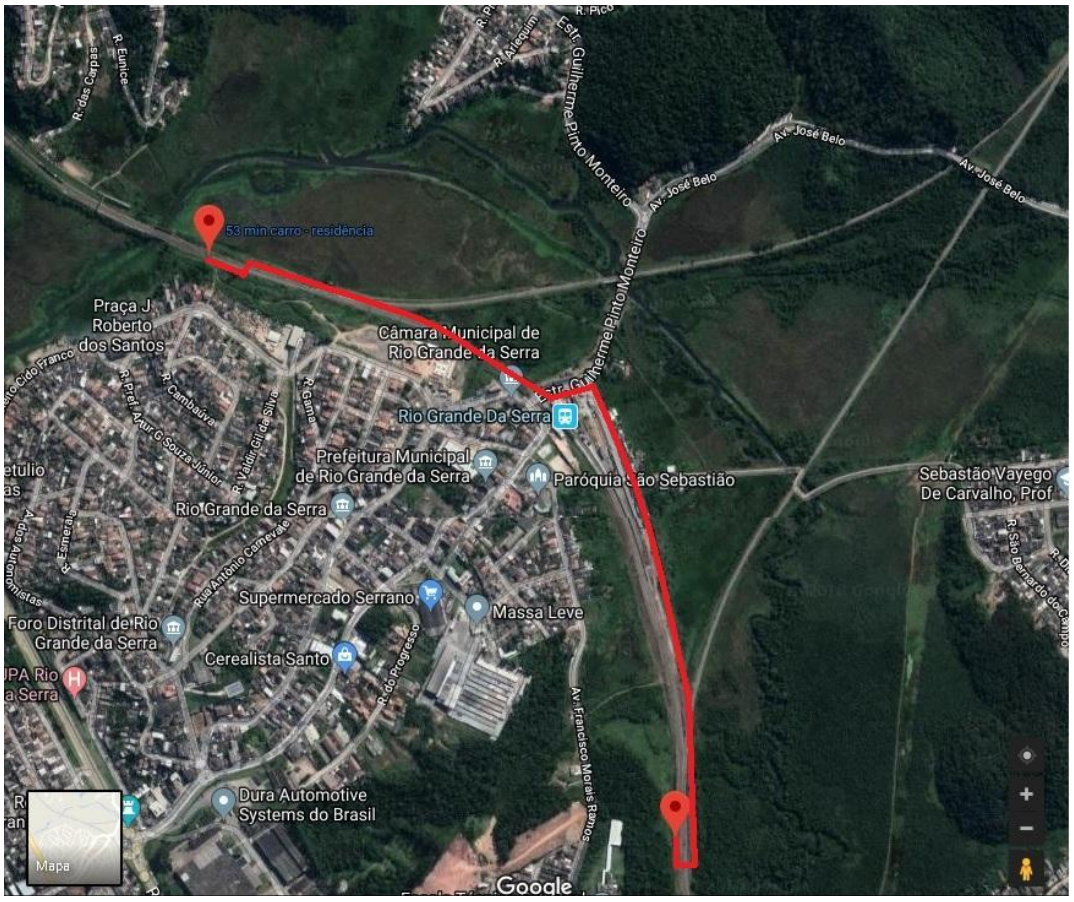


Figura 3: Encaminhamento da tubulação subterrânea para passagem dos cabos (Fonte: Google Maps)

O terminal é o ponto final da linha, onde o trem deve chegar, parar e retornar. Por esse motivo, esses bangalôs são responsáveis pelas informações de um ponto importante da linha, pois são eles que indicam quando o trem está preparado para mudar de sentido e iniciar uma nova viagem.

2.2 Sistema Atual

Para visualizar os equipamentos instalados na via é utilizado um diagrama, chamado de *track plan* ou plano de via em português (Figura 4). Com ele também é possível identificar por números cada equipamento, que veremos mais à frente, serão usados para associar o dado transmitido no canal ao equipamento.

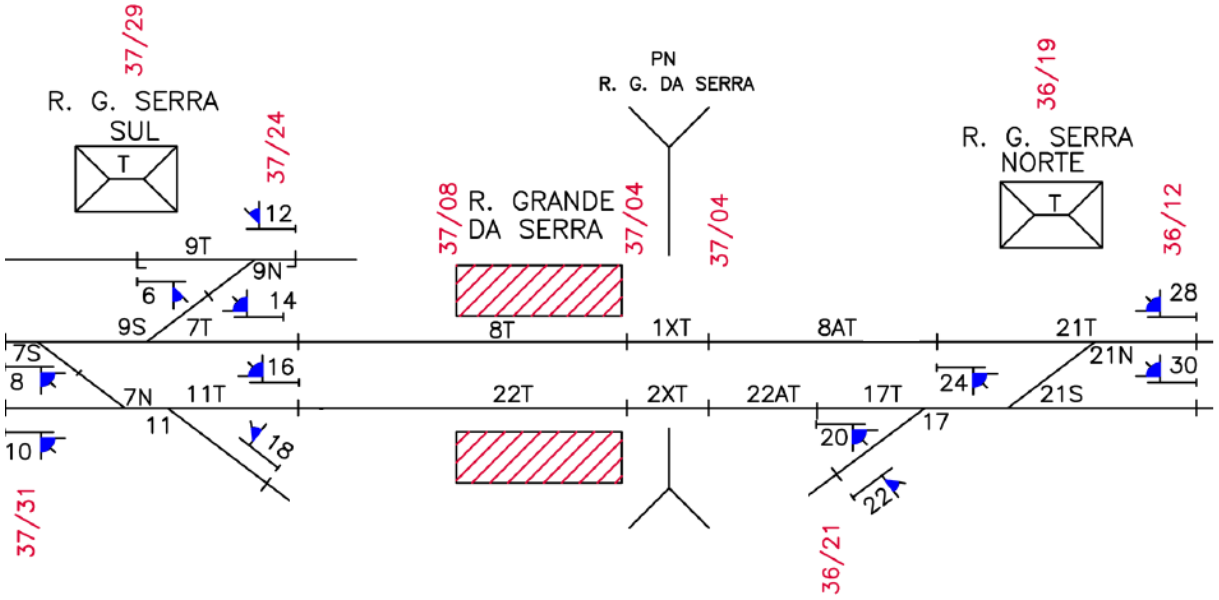
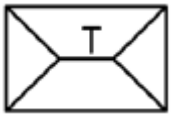
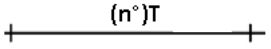
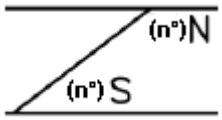
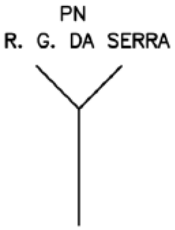


Figura 4: Plano de via de Rio Grande da Serra

Tabela 2: Legenda do plano de via

	Plataforma da estação
	Bangalô (local de alvenaria para abrigar relés dos circuitos de sinalização)
	Bloco do circuito de via (T = track, n° = identificação)
	Sinal 3 aspectos (verde, amarelo ou vermelho, n° = identificação par)

	Aparelho de mudança de via (N = north, S = south, n° = identificação ímpar)
	Passagem em Nível, cruzamento entre ferrovia e rodovia
	Localização (XX = quilômetros de distância do marco zero na Luz, YY = contagem do poste da rede aérea)

No sistema atual existem 17 cabos responsáveis por levar os dados de indicação que funcionam de forma paralela (Figura 5) com dois símbolos distintos, nível alto (16V) e nível baixo (0V).

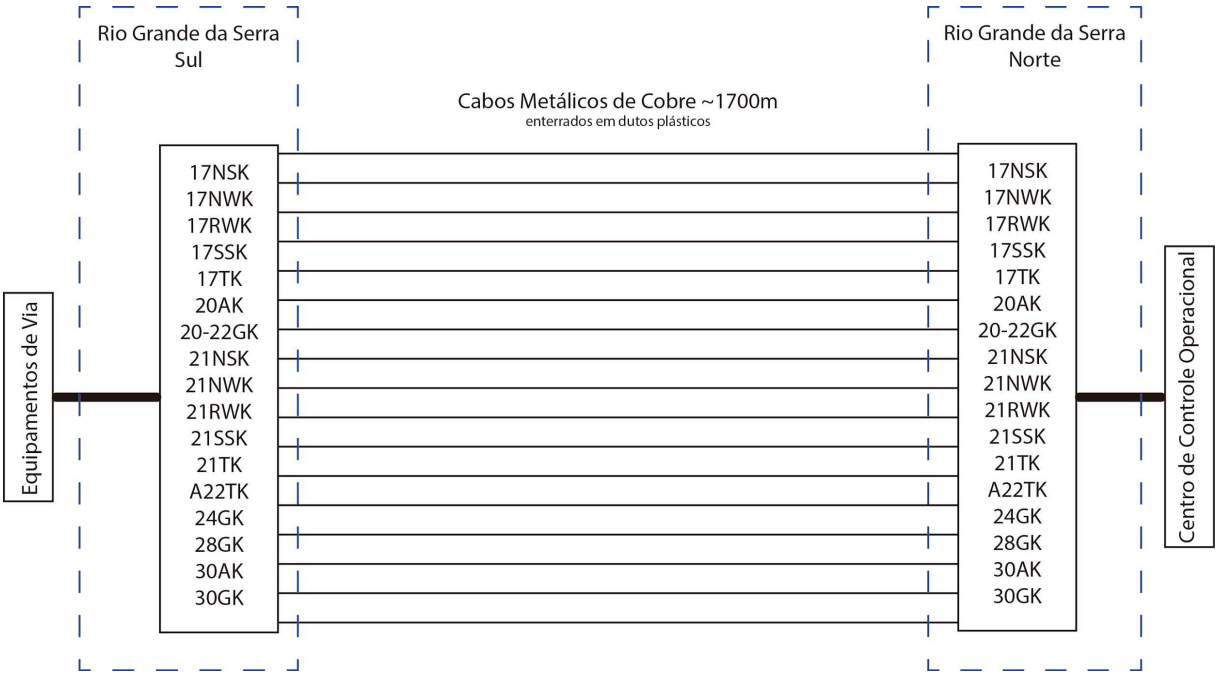


Figura 5: Diagrama de blocos do sistema de transmissão de dados RGS Sul para o Norte

Descrição dos dados transmitidos:

- TK - Indica se o trem está ou não em determinada posição;
- GK - Indica a condição do sinal (aberto ou fechado);
- NSK ou SSK - Indicam o sentido liberado para circulação do trem;
- NWK - Indica se a máquina de chave está posicionada para a reta;
- RWK - Indica se a máquina de chave está posicionada para trocar o trem de via.
- AK - Indica se o trecho na aproximação do sinal está liberado para circulação.

2.3 Proposta de Alteração

Assim, o grande desafio deste estudo, que resultará no projeto de substituição dos cabos metálicos, será receber as informações paralelamente, processá-las e transmiti-las de forma serial, ou seja, enfileirando todos os dados bit a bit em forma de luz, pela fibra óptica agregadas em um protocolo TDM (time division multiplexing).

A partir disso, na outra ponta, deverá ser feito o processo contrário, recebendo os dados serialmente, processando e convertendo corretamente em paralelo, conforme o diagrama de bloco apresentado na Figura 6.

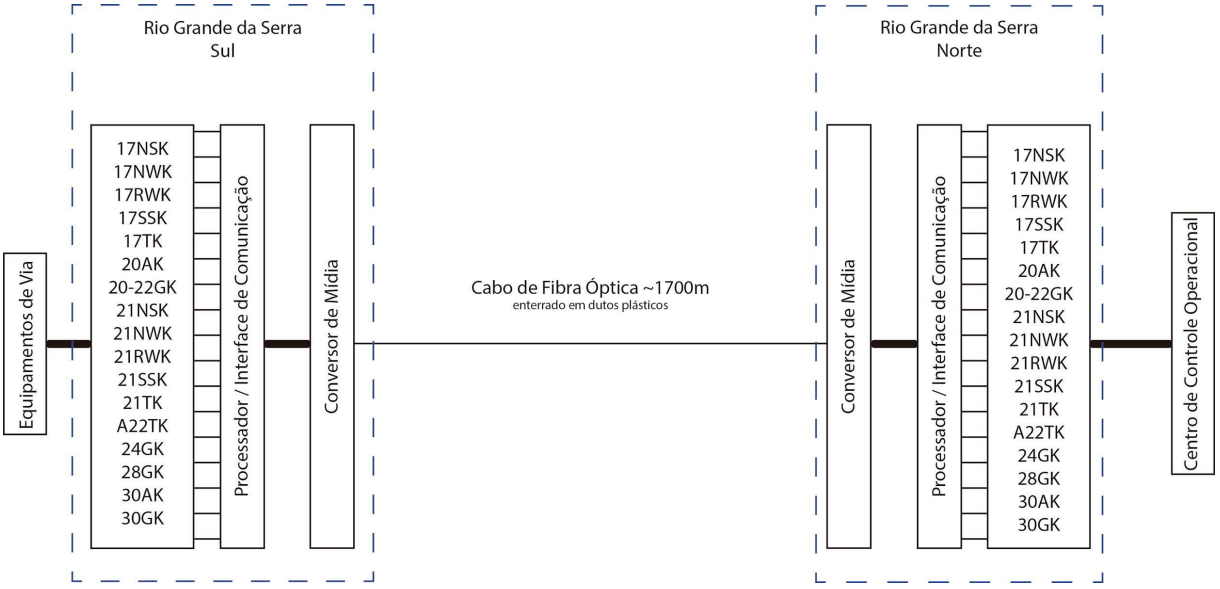


Figura 6: Diagrama de blocos do sistema proposto de RGS Sul para o Norte

Como visto, o processo completo de transmissão dos dados gerados nos equipamentos de via e transmitidos até o Centro de Controle Operacional (CCO) necessita passar por 5 etapas, que aqui serão chamadas de módulos:

- Módulo de Entrada

Nesse módulo constará uma placa responsável pela conversão do nível de tensão do sistema atual, que é de 16V para 5V, nível de tensão utilizado pelo processador. Além de isolar eletricamente o novo sistema de comunicação dos equipamentos já existentes, evitando que surtos se propaguem pelos circuitos mais vulneráveis.

- Módulo de Saída

Os módulos de saída possuem função contrária ao de entrada, são controlados pelo processador com nível de tensão em 5V, que a partir de circuitos de relés, acionam o sistema antigo com 16V, conforme mensagem recebida através do canal óptico.

- Módulo Processador

Este módulo é aplicado nos dois extremos do sistema (Rio Grande Norte e também no Sul), na primeira executa todo o processo de multiplexação dos dados recebidos paralelamente, montando *frames* em um protocolo próprio e passando para o conversor de mídia.

Já no lado oposto o processo é inverso, ele demultiplexa os dados vindos em frames para que o módulo de saída atue corretamente e alimente os equipamentos do sistema.

- Módulo Conversor de Mídia

Como o próprio nome já diz, o conversor de mídia é responsável por converter os bits recebidos em pulsos elétricos para intensidade luminosa e também o processo contrário.

- Módulo do Canal

A fibra óptica é o canal de comunicação do sistema passada em dutos subterrâneos e conectada aos conversores de mídia nos dois extremos, que neste caso será de 1700m de distância.

3. Construção do Protótipo

Como dito nos objetivos deste trabalho, para exemplificar a proposta de melhoria do sistema, será construído um protótipo, simulando a coleta e transmissão de 8 bits de informação. Este, assim como o sistema proposto é dividido por partes e cada parte executa uma função específica do projeto, que serão descritas e detalhadas abaixo:

3.1 Circuito simulador de entrada dos dados

Anteriormente, no item 2.3, foi apresentado que a tensão de trabalho dos equipamentos de via, tanto controle como indicação, são de 16V nominais, mas devido a dificuldade de encontrar dispositivos eletrônicos neste nível de tensão comercialmente, decidiu-se adotar 12V.

Sendo assim, para simular as indicações dadas pelos equipamentos de via foi construído um circuito (Figura 7) com chaves manuais que enviam um sinal de 12V ou 0V para o módulo processador, simulando um bit binário.

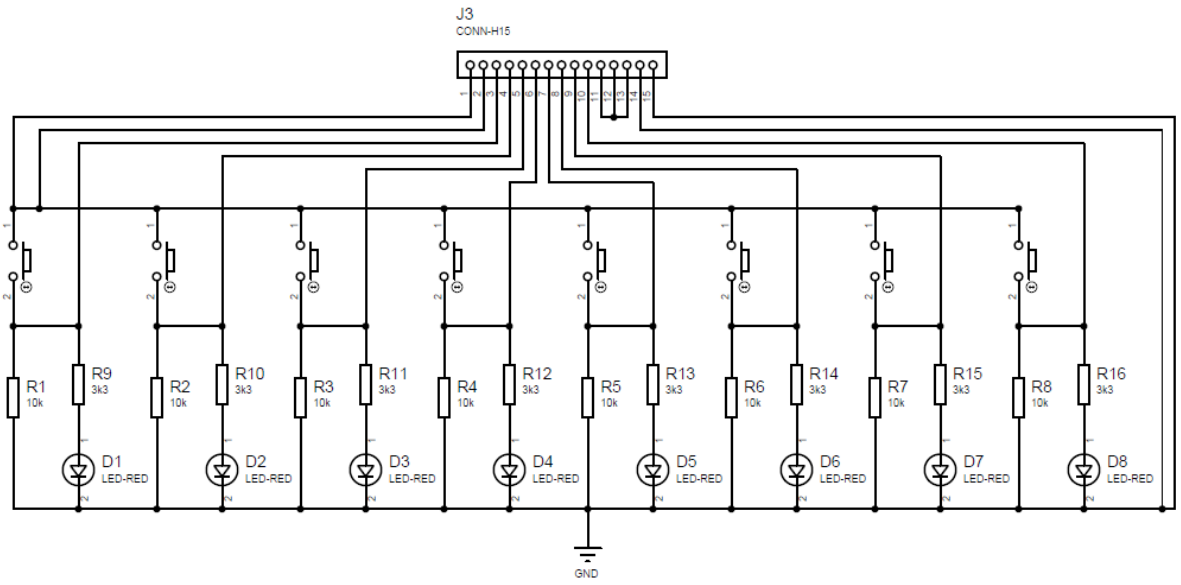


Figura 7: Circuito simulador das entradas do sistema.

3.2 Circuito de Interface de Entrada

Como o processador utilizado efetua a leitura das portas de entrada com tensão em 5Vcc, foi necessário desenvolver um circuito de interface, identificando o nível alto do simulador de entrada (12Vcc) e convertendo-o para o nível mais baixo.

Para isto, utilizou-se o chip 4N25, conhecido também como um optoacoplador (Figura 8).

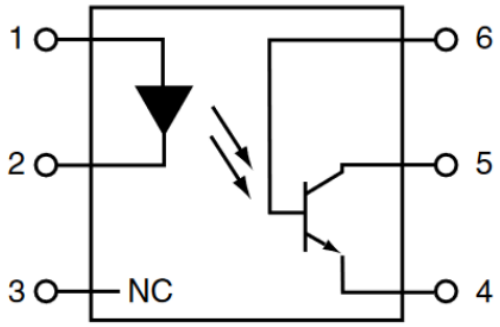


Figura 8: Esquema representativo do chip 4N25 Fairchild.

Este dispositivo possui um diodo emissor de luz e um fototransistor, para utilizá-lo basta alimentar o diodo pelos pinos 1 e 2 do chip e conectar o transistor no circuito para trabalhar na configuração de chave eletrônica entre os terminais 4 e 5.

Como no protótipo serão simulados 8 entradas são necessários, portanto, 8 acopladores (Figura 10).

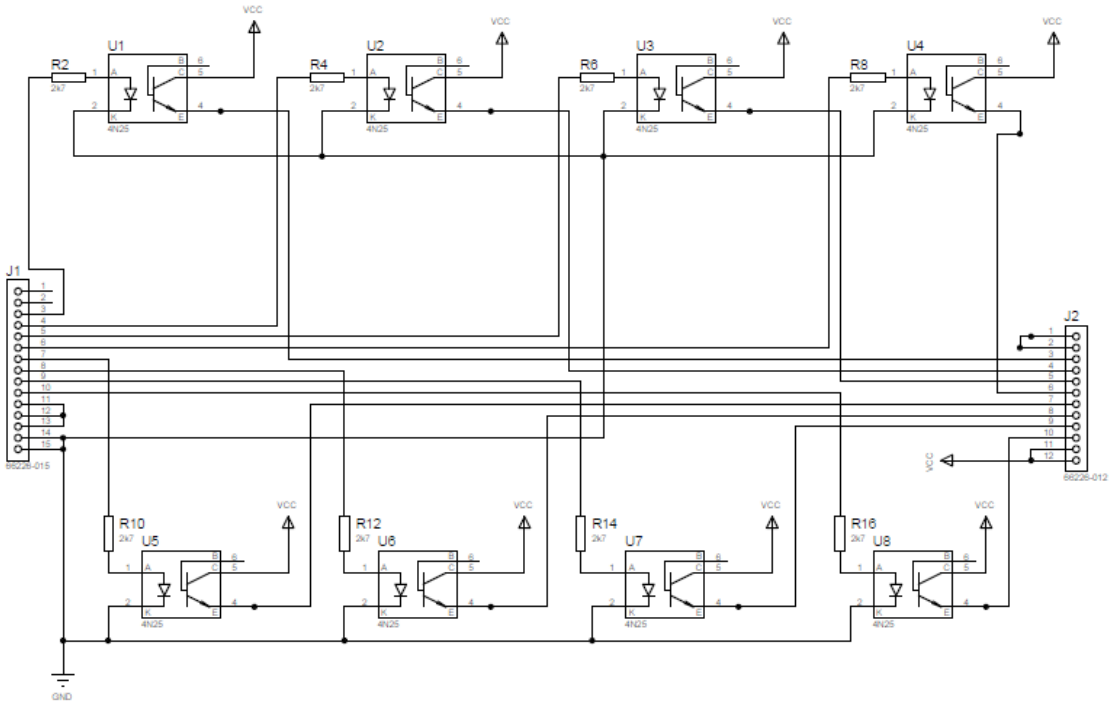


Figura 10: Esquema elétrico do circuito de interface 12V para 5V.

Neste circuito, a entrada dos sinais do simulador de entrada é feita pelo conector J1, passando pelo optoacoplador tem-se a conversão para 5V na qual é conectada ao processador pelo conector J2.

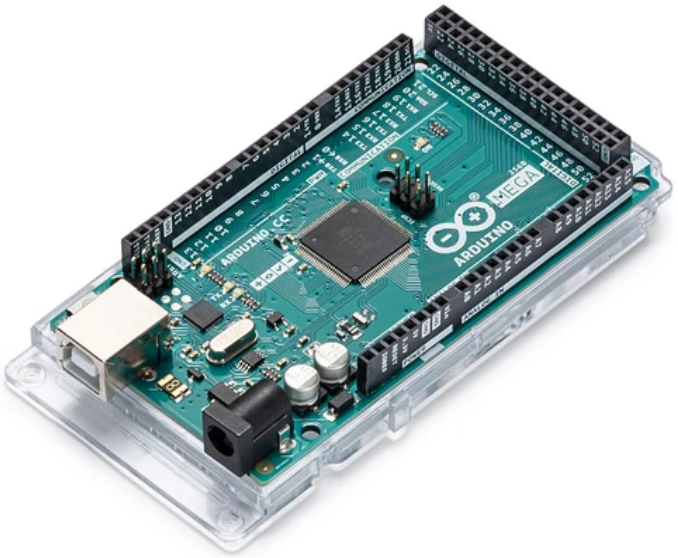
3.3 Circuito processador do transmissor

Esta etapa compõe a parte principal do projeto. Neste momento, os sinais foram recebidos de forma paralela e deverão ser manipulados e preparados para o envio serial no canal óptico.

O dispositivo escolhido para executar esta tarefa foi o Arduino, por ser uma placa eletrônica de código e hardware aberto, de baixo custo e de fácil interação com outras plataformas e protocolos.

3.3.1 Arduíno

Esta placa foi desenvolvida no ano de 2005 na região de Ivrea, Itália para ser uma ferramenta simples e fácil de prototipagem eletrônica. Ele é composto basicamente por um controlador Atmel AVR de 8 bits, uma interface serial USB e algumas portas digitais e analógicas de entradas e saídas de informações (Figura 11)



.

Figura 11: Foto da placa Arduino (Fonte: site da loja oficial arduino.cc).

Devido ao fato de ser de código e hardware abertos, ou seja, todos os dados da construção física e do software são disponíveis a qualquer pessoa e permitem alterações e distribuições independentes, este dispositivo ganhou diversas melhorias e capacidade de interação com várias tecnologias.

E esta capacidade foi o fator determinante para a escolha deste dispositivo no projeto. Pois, mais a frente será exibido com detalhes que o conversor de mídia utilizado necessita que os dados sejam inseridos com protocolo Ethernet, o que é facilmente possível por meio de uma placa chamada Shield Ethernet, que acopla na placa do Arduíno perfeitamente (Figura 12).

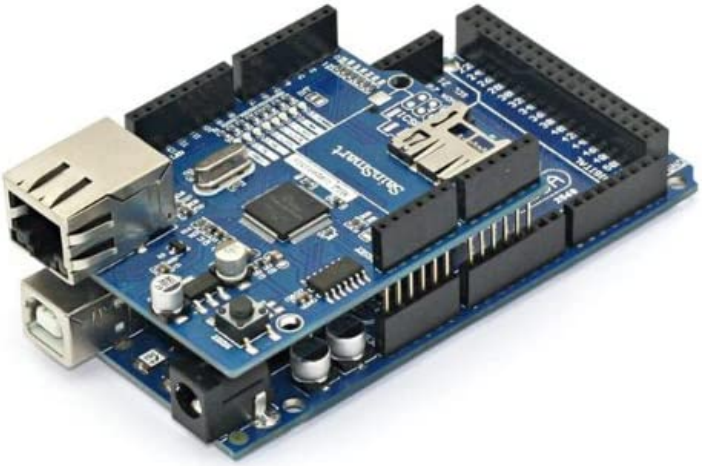


Figura 12: Shield Ethernet conectada sobre o Arduino (Fonte: loja da amazon.com).

Depois dos dados coletados pelas entradas digitais do Arduino passa-se a etapa de tratamento dos dados antes da transmissão, que será descrita abaixo.

3.3.2 Estrutura do software transmissor

Realizar a leitura das entradas é apenas o começo do processo de tratamento dos dados no lado transmissor. A ideia nesta etapa é receber estas informações, codificá-las para diminuir a probabilidade de erros e até corrigi-los, estruturar frames e empacotar em um determinado protocolo para serem transmitidos.

Como cada equipamento de via é um bit de informação e neste trecho abrangido pelo projeto contempla 17 dispositivos, seriam necessários 17 bits sendo codificados e empacotados ao mesmo tempo para transmissão. Isto poderia exigir uma capacidade de processamento muito grande do sistema, por isso, os dados foram estruturados da seguinte forma:

As entradas serão divididas em grupos de 4, cada grupo será chamado aqui de placa, ou seja, teremos a placa de nome 1 (em binário 0001), com suas respectivas 4 entradas, depois haverá outra placa chamada 2 (em binário 0010), com outras 4 entradas e assim por diante, até a placa 15 (Figura 13).

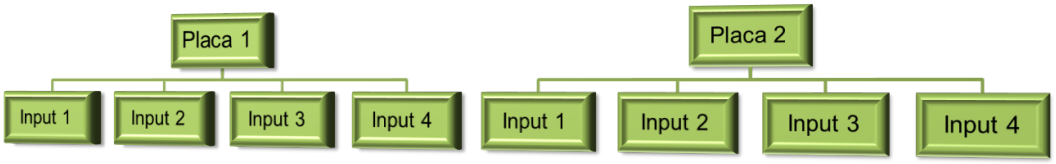


Figura 13: Estrutura dos dados de entrada.

Desta forma, como cada placa é endereçada com 4 bits e a primeira sendo a número 1, temos um limite de 15 placas e como cada uma possui 4 entradas, o sistema terá capacidade de 60 entradas no total.

Com esta estrutura, pôde-se elaborar um software que demandaria menos capacidade de processamento, exigindo menos recursos do sistema para a codificação.

Agora, o único problema é como o receptor vai saber identificar se a mensagem recebida contém o endereço da placa ou se é referente ao estado das entradas. Para resolver isto, foi criado uma espécie de *flag,* de apenas um bit, que se 0, indica que é o endereço da placa e se 1, indica que se refere ao estado das entradas.

Sendo assim, haverão dois tipos de *frames* diferentes, o primeiro com o endereço da placa e o segundo com o estado das saídas. Apesar disto, a estrutura de montagem do dois é a mesma, conforme fluxograma abaixo (Figura 14):

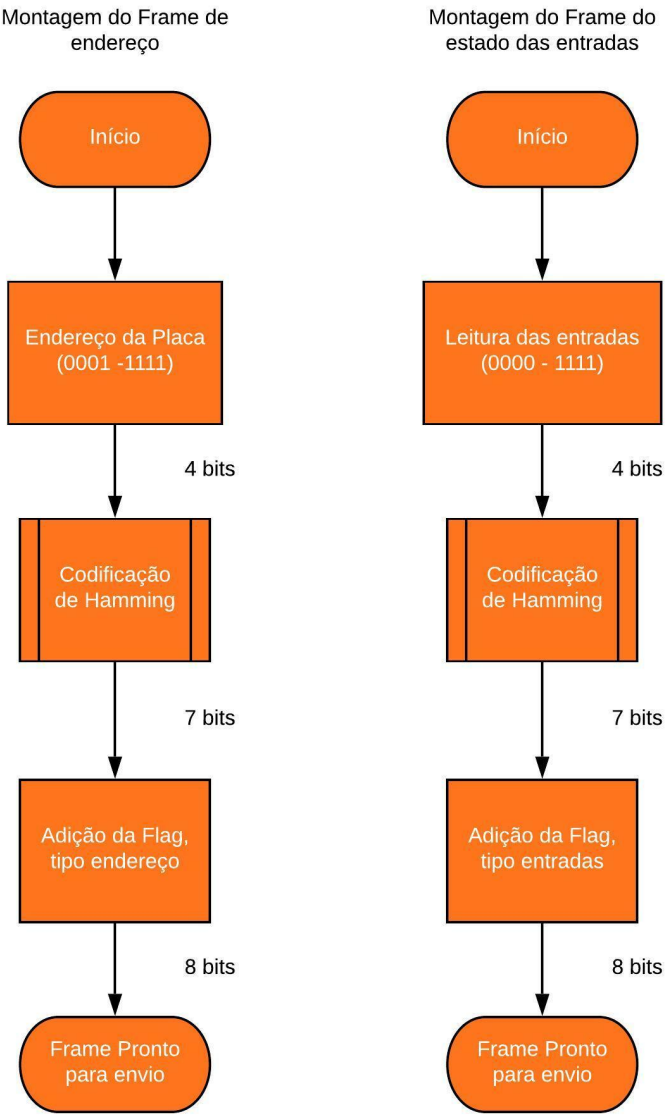


Figura 14: Fluxograma da montagem dos *frames*.

Agora o sistema irá varrer uma placa de cada vez, enviando primeiro o endereço e depois, num segundo *frame*, o estado das entradas, repetindo este processo infinitamente.

É possível ver todas as tarefas executadas pelo processador do lado transmissor, exemplificado no fluxograma abaixo (Figura 15).

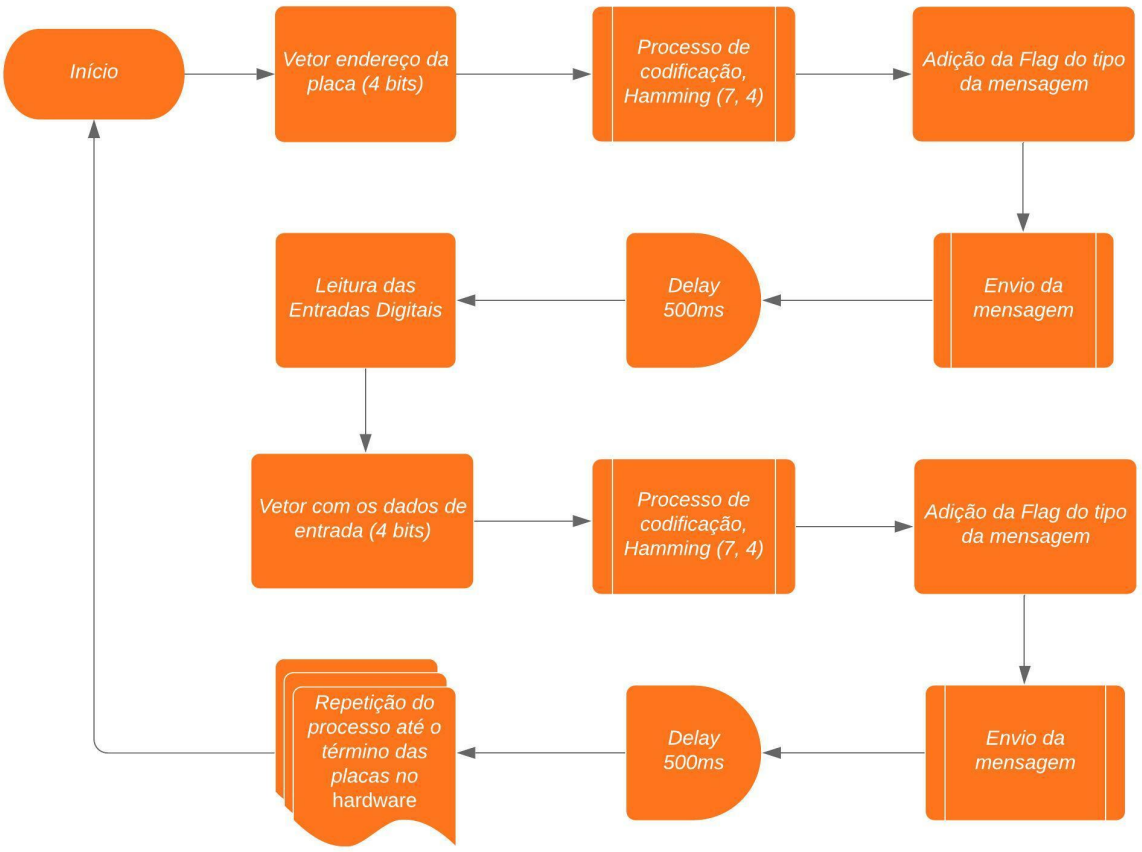


Figura 15: Fluxograma do transmissor.

Como visto, este é um sistema unidirecional em que o transmissor não recebe em nenhum momento confirmação do lado receptor se a mensagem está sendo entregue corretamente ou não. E não há nenhum problema em ser assim, dado que o sistema ferroviário não demanda alta velocidade de transmissão de dados, podendo esperar o próximo ciclo de varredura para enviar corretamente os dados, caso o anterior esteja com erro.

No processo de codificação indicado nas figuras acima, foi mencionado o nome Hamming, este é um processo matemático de codificação de canal e merece um capítulo dedicado.

3.3.3 Codificação de Hamming

Este tipo de código é utilizado para atender duas funções do projeto. A primeira é para codificar os dados enviados e a segunda é para detectar e corrigir erros que possam ocorrer na transmissão.

Esta codificação foi desenvolvida pelo cientista Richard W. Hamming que trabalhava no Laboratório Bell de Tecnologia, que ficava furioso com os computadores da década de 40 que ao se depararem com erros nos arquivos (que na época eram gravados em cartões perfurados) parava de processar os dados.

Ele questionou a capacidade desses computadores de apenas detectar o erro, mas não conseguiam corrigi-los. Desta forma ele desenvolveu essa teoria.

Definições:

- Distância mínima é a menor quantidade de bits que pode variar uma mensagem da outra.
- Capacidade de correção: $k = \frac{d-1}{2}$, na qual d é a distância mínima e k é a quantidade de erros corrigíveis.

Corolário:

- Um código C pode corrigir até k erros se e somente se sua distância mínima d(C) verifica a desigualdade: $d(C) \geq 2k+1$.

No projeto, foi escolhido este tipo de codificação, pois o canal de comunicação é fechado (duas extremidades da fibra óptica) e as operações matemáticas não são tão complexas, facilmente realizáveis pelo processador utilizado. Além de conseguir corrigir 1 bit caso haja um erro ou pelo menos detectar que a mensagem está errada se houverem mais erros.

A matriz geradora, principal operador da codificação é composta parte por dados de paridade e a outra parte por uma matriz identidade que replica a mensagem.

$$G = | P : I_k |$$

$$\left|\begin{array}{ccc:cccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right|$$

O vetor de 4 *bits* recebido é multiplicados por esta matriz e depois feito uma operação de ou exclusivo entre eles. Conforme operação abaixo:

$$C = \left|\begin{array}{cccc} m1 & m2 & m3 & m4 \end{array}\right| \times \left|\begin{array}{ccc:cccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right|$$

Na qual, o vetor m é a mensagem e C é a mesma mensagem, mas agora codificada. Como se vê, entram 4 bits (k) na operação e saem 7 (n). Os 3 *bits* a mais são justamente os *bits* de paridade que auxiliam na detecção de erros. E assim dizemos que será uma codificação (7,4).

No *software*, esta operação é feita com auxílio de duas rotinas *for*, como mostrado a seguir (Figura 16):

```
134   for (int j = 0; j<7; j++){
135     for (int i = 0; i<4; i++){
136       bool aux = vet1[i] & G[i][j];
137       x[j] = x[j] ^ aux;
138     }
139     send = send + (x[j] << (7-j));
140   }
```

Figura 16: Trecho do código de codificação da mensagem.

3.3.4 Adaptador Ethernet

Depois de realizado a codificação das mensagens, o *frame* agora composto por 8 *bits*, é enviado, como dito anteriormente, para uma *shield Ethernet* que é

conectada diretamente ao processador, este módulo é responsável pelo empacotamento dos dados em um protocolo UDP.

Ele é composto por um chip W5500 capaz de organizar as informações enviadas pelo processador em diversos protocolos de comunicação TCP/IP, como o TCP, IPv4, ICMP, ARP, IGMP, PPPoE e o próprio UDP.

Ao fim do processamento, o chip envia para um terminal do tipo RJ45 a mensagem, que pode ser conectado a qualquer rede estruturada padrão Ethernet. A estrutura desta placa é exibida abaixo (Figura 17):

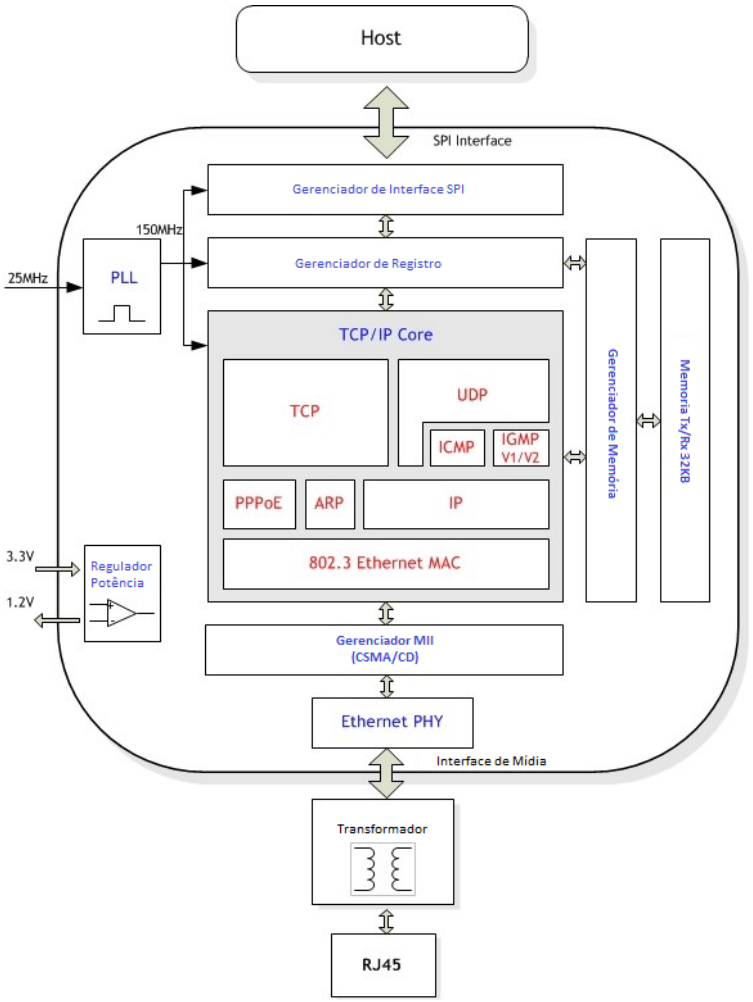


Figura 17: Diagrama de blocos do chip W5500 gerador do protocolo UDP.

O protocolo UDP foi escolhido devido a não necessidade de confirmação das mensagens recebidas. Como a varredura das entradas será realizada o tempo todo, caso haja uma perda de pacote, basta esperar o próximo ciclo para receber a

atualização da condição atual. Este protocolo é descrito com mais detalhes no capítulo seguinte.

3.3.5 Protocolo UDP

Este protocolo é considerado o modelo mais simples em uma conexão de rede na camada de transporte. Ele não executa nenhum controle de congestionamento da troca de dados entre cliente e servidor, também não verifica se o pacote chegou ao destinatário, podendo tolerar até alguma perda [8].

Sua estrutura é bem simples, conforme visto abaixo (Figura 18), possui o número das portas da origem e do destino, o tamanho do pacote de dados, um verificador de erros chamado de *checksum* e a informação que foi transmitida [8].

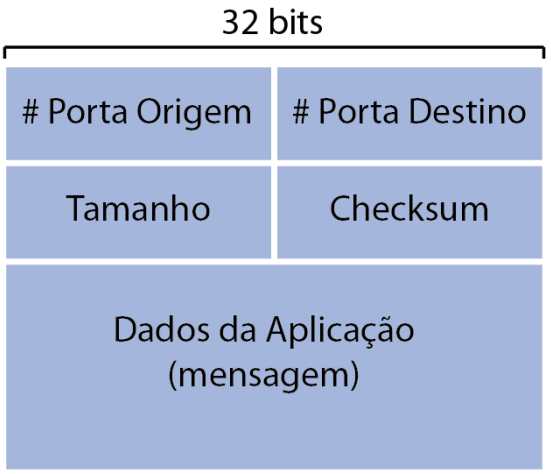


Figura 18: Estrutura de mensagem no protocolo UDP

O *checksum* é uma técnica utilizada para a detecção de erros, neste caso, durante a verificação, todas as palavras de 16 bits são somadas, ao fim de todas as somas faz-se o complemento de 1 dessa palavra, obtendo-se o valor do *checksum*, conforme abaixo:

Tabela 4 : Operação de Checksum

Exemplo de Resultado final da soma [A]	1010111001111010
Complemento de 1 (checksum) [B]	0101000110000101
Verificação do Resultado [A + B]	1111111111111111

Se, na verificação do resultado, algum bit for 0, significa que houve algum erro na transmissão e o pacote todo é descartado.

3.4 Conversor de Mídia

Este dispositivo fabricado pela empresa Planet, do modelo FT-802S15 (FIgura 18), possui um circuito que recebe dados por pulsos elétricos, via cabos metálicos e converte em pulsos de luz, que são transmitidos por cabos de fibra óptica.

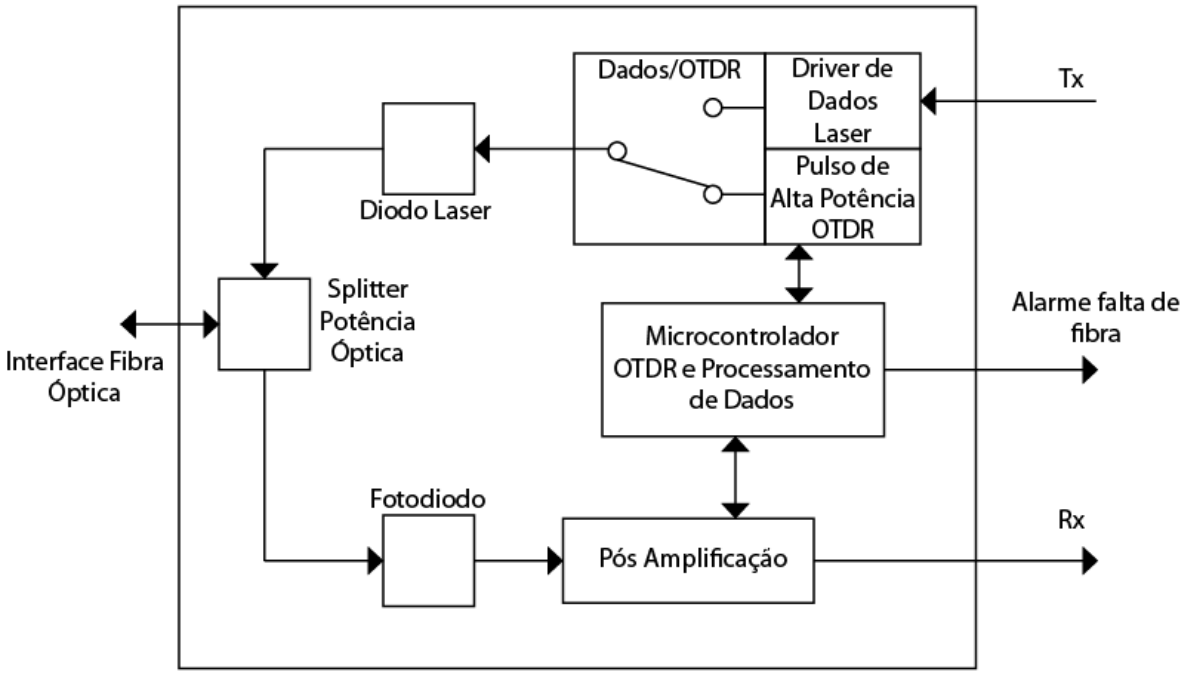


Figura 18: Diagrama de blocos do conversor de fibra óptica para Ethernet

A interface de comunicação cabeada neste modelo de conversor é do tipo Fast Ethernet 10/100Mbps RJ-45 e o plugue da fibra óptica é do modelo SC (Figura 19).

Figura 19: Foto do conversor de mídia FT-802s15 da marca Planet

Tem capacidade para transmissão de até 15km de distância entre os módulos e comprimento de onda à 1310 nm.

No projeto então, são colocados dois dispositivos deste modelo. Um do lado transmissor e outro do lado receptor, fazendo o link de fibra óptica entre as duas partes.

3.5 Circuito processador do receptor

Após os dados chegarem ao conversor de mídia, ele reconverte para o protocolo *Ethernet/UDP* conectado com um cabo RJ45 na *shield Ethernet* do Arduino. O processador do receptor tem a mesma estrutura do transmissor (conversor de mídia → shield Ethernet → Arduino), a diferença está no software que será descrito no capítulo seguinte.

3.5.1 Estrutura do software receptor

Quando o *frame* chega ao processador, depois de ser separado do UDP, possui a estrutura de 8 bits, sendo 7 bits da informação codificada e 1 bit com o identificador do tipo da mensagem (Figura 20).

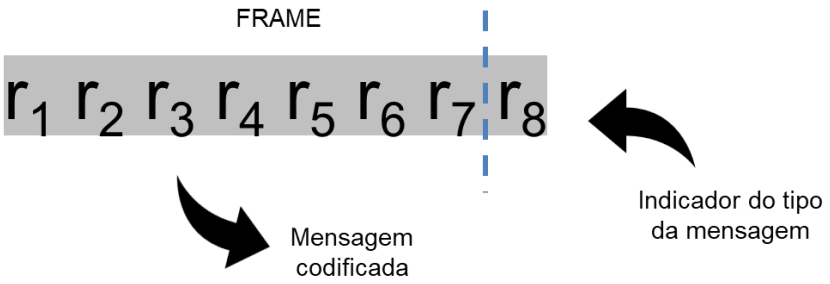


Figura 21: Estrutura do frame recebido, 1 byte.

Se o tipo da mensagem for igual a 0, o software decodifica a mensagem, armazena o endereço da placa e aguarda o próximo frame com os dados das condições das 4 saídas correspondentes desta placa, que deverá vir com o indicador do tipo igual a 1.

Caso o receptor receba duas mensagens seguidas com indicador igual a 0, subtende-se que um pacote foi perdido neste intervalo, e estes dados são descartados aguardando o próximo ciclo de varredura.

E, se recebido duas mensagens seguidas com indicador igual a 1, também tem-se o mesmo entendimento, descartando a última mensagem até que chegue corretamente uma mensagem do tipo endereço de placa seguida por uma de acionamento das saídas.

É possível ver todo este processo com auxílio do fluxograma do receptor abaixo (Figura 22).

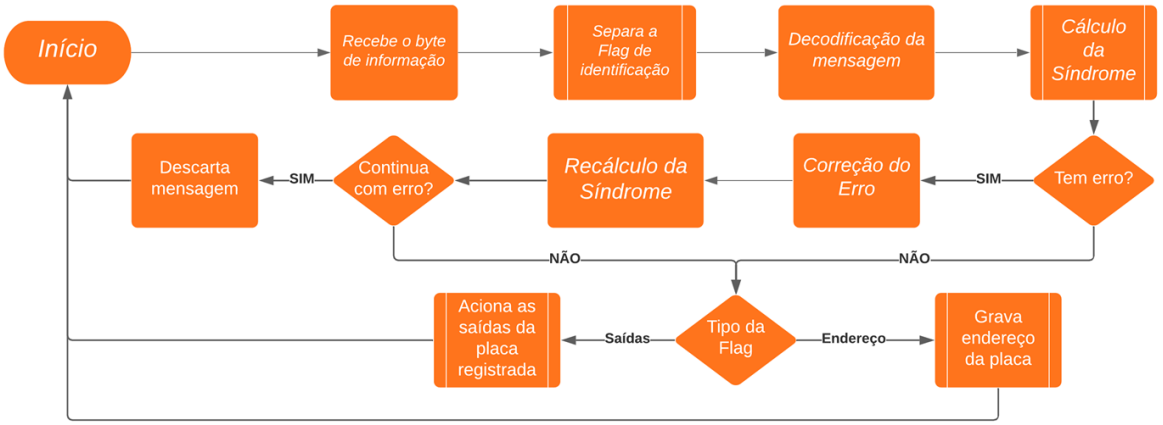


Figura 22: Fluxograma do receptor.

Neste esquema a palavra síndrome aparece algumas vezes e faz parte da decodificação da mensagem que é complementar da operação de Hamming.

3.5.2 Decodificação de Hamming

Para decodificar os 7 *bits* recebidos, será necessário utilizar uma nova matriz, chamada H, que é construída a partir da matriz geradora de codificação. Ela é composta por uma matriz identidade mais a matriz de paridade utilizada na codificação, só que transposta (Figura 23).

$$H = | I_{n-k} : P^T |$$

$$H = \left| \begin{matrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{matrix} \right|$$

Figura 23: Matriz de decodificação.

Quando realizada a operação de multiplicação e soma exclusiva entre o vetor da mensagem recebida (r) por esta matriz H transposta, obtém-se o valor da síndrome (S) (Figura 24).

$$S = r \times H^T$$

$$\begin{vmatrix} r1 & r2 & r3 & r4 & r5 & r6 & r7 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{vmatrix} = S$$

Figura 24: Operação de decodificação.

Se o valor de S for igual a 0 significa que a mensagem recebida é válida, sendo possível extrair a mensagem do vetor r. Caso contrário, significa que existem erros. Para cada valor de S, atribui-se uma posição do vetor que está com este erro, sendo possível construir uma tabela de identificação da posição errada.

Tabela 5 : Posição do erro e valor da síndrome associada

e							$S = e \times H^T$		
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	1
0	0	0	1	0	0	0	1	1	0
0	0	0	0	1	0	0	0	1	1
0	0	0	0	0	1	0	1	1	1
0	0	0	0	0	0	1	1	0	1

Durante o processo de decodificação, faz-se a conta da síndrome a primeira vez, caso haja um erro, o *software* corrige o bit identificado e faz novamente a operação do cálculo da síndrome, nesta segunda vez, caso for encontrado um novo erro, a mensagem deve ser descartada, pois nesta configuração pode-se corrigir apenas um bit, apesar de ser possível detectar mais de um erro.

Passado por todo este processo, a mensagem pode enfim chegar às saídas digitais do Arduino, correspondente à placa endereçada anteriormente.

3.6 Circuito de Interface de Saída

Como as saídas do módulo de processamento são de baixa potência, faz-se necessária uma interface para acionamento das cargas. Neste projeto, a forma utilizada para isso foi com auxílio de um relé de contato seco (Figura 25).

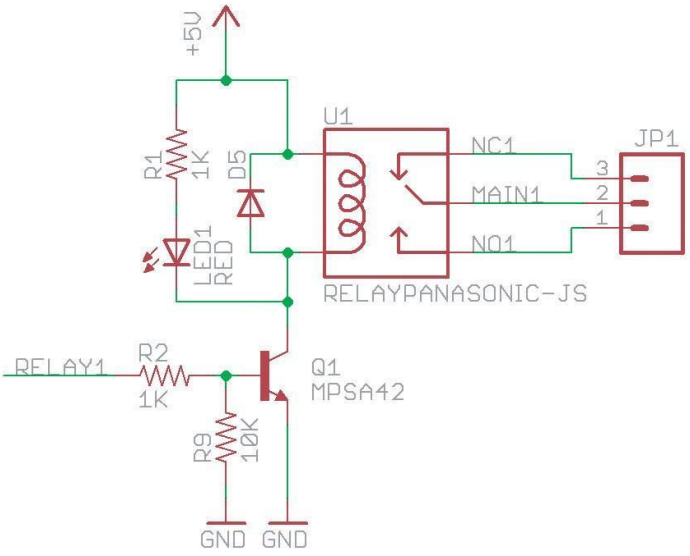


Figura 25: Circuito de interface de saída à relé

O Arduino alimenta a base de um transistor NPN que permite ao relé a mudança de estado dos seus contatos metálicos, nomeado como terminal JP1. Este, por sua vez, pode acionar cargas que consomem até 10A de corrente, mais que o suficiente para a aplicação deste projeto.

3.7 Circuito simulador de saídas dos dados

Para facilitar a visualização dos dados transmitidos, foi projetado um painel com indicadores luminosos. Cada saída é ligada a um LED que indica o acionamento da saída (Figura 26).

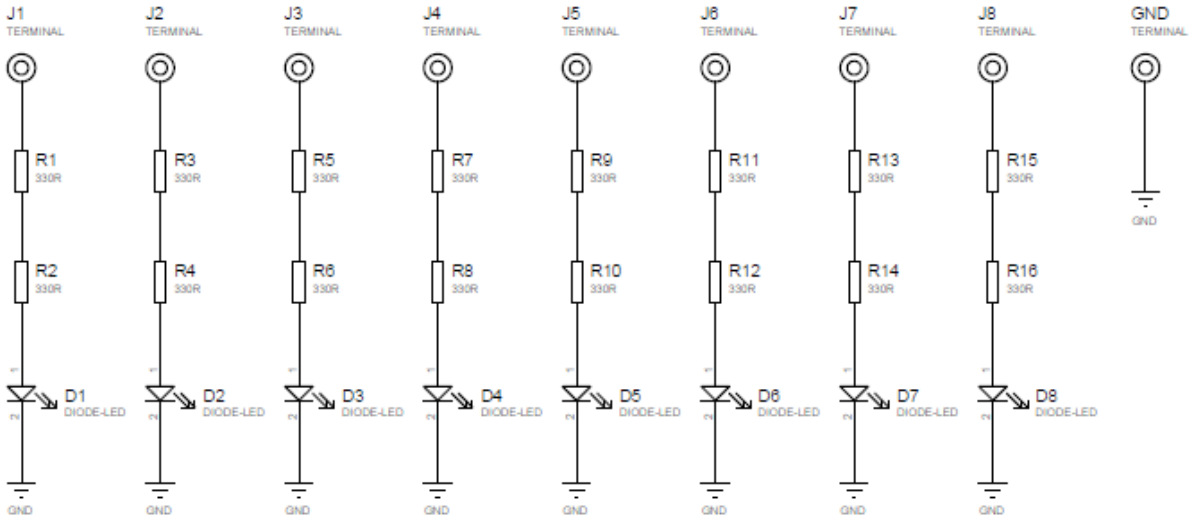


Figura 26: Circuito elétrico do simulador de saída de dados

4. Resultados

Nesta etapa serão detalhados e exibidos (Figura 27) toda a lógica de montagem do protótipo e como eles interagem.

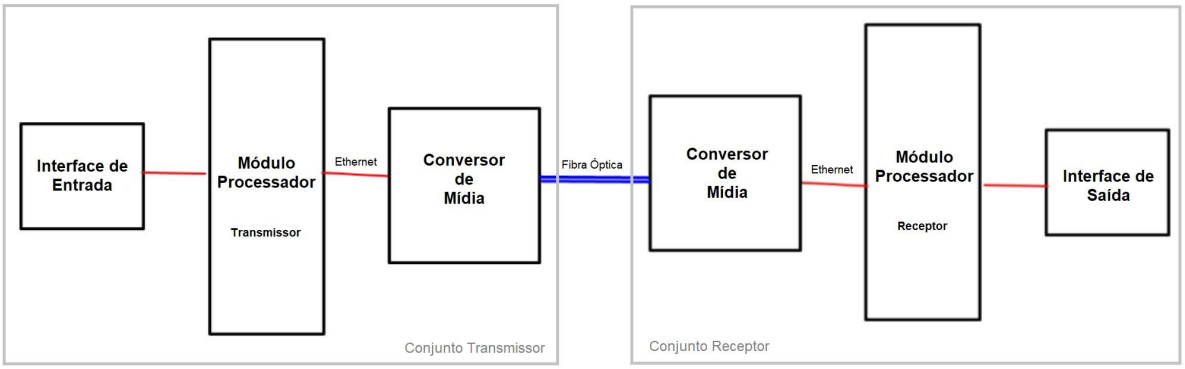


Figura 27: Esquemático de montagem do protótipo.

A interface de entrada é composta por 8 chaves (Figura 28) que simulam as indicações dos equipamentos instalados ao longo da ferrovia. Na sua conexão com o módulo processador ela apenas liga 12V ou 0V em cada entrada.

Figura 28: Interface de entrada

No módulo processador transmissor, foi colocado um circuito acoplador, baseado no CI 4N25 para detectar os bits enviados pela interface de entrada, o próprio Arduíno que executa toda a função lógica de varredura das portas de entrada e criação dos pacotes de dados a serem enviados acoplado ao módulo Ethernet que envia estes pacotes para o próximo estágio em protocolo UDP. Todos estes componentes foram montados em uma caixa plástica (Figura 29).

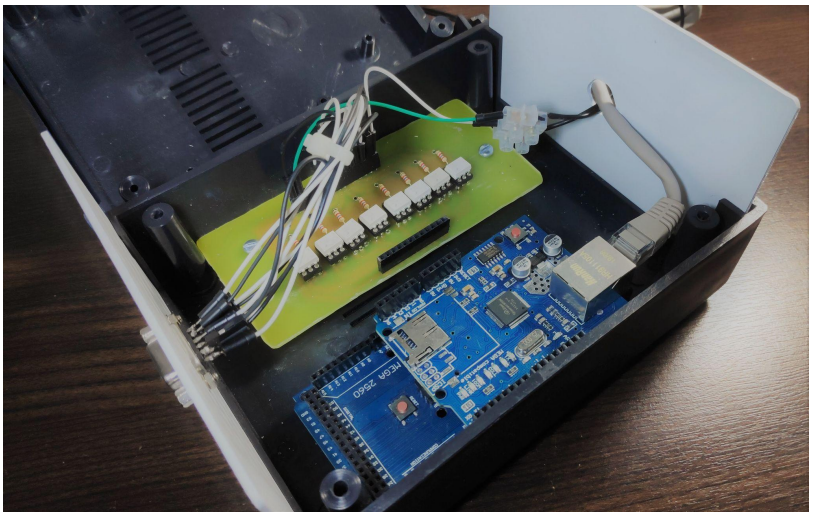


Figura 29: Construção do módulo processador transmissor - Processador Arduino com shield ethernet e uma PCI de interface com acopladores

O conversor de mídia recebe estes dados e possibilita a alteração do meio físico, passando de um sinal elétrico para um sinal luminoso, encaminhado pela fibra óptica.

O processo contrário ocorre do lado do conjunto da recepção, na qual os sinais luminosos são convertidos em sinais elétricos, passando os dados via protocolo UDP para o módulo processador receptor.

Os conversores são vendidos em pares (Figura 30). Só assim para funcionarem corretamente.

Figura 30: Conversores de mídia comerciais.

No módulo processador de recepção há novamente uma interface *Ethernet* conectada ao processador, que realiza toda a operação de decodificação e verificação de erros das mensagens recebidas. Caso a mensagem seja validada, as saídas são acionadas, que no caso, é um conjunto de relés magnéticos simples, fornecendo 12V ou 0V nas saídas. Todo este conjunto também foi montado em uma caixa plástica (Figura 31).

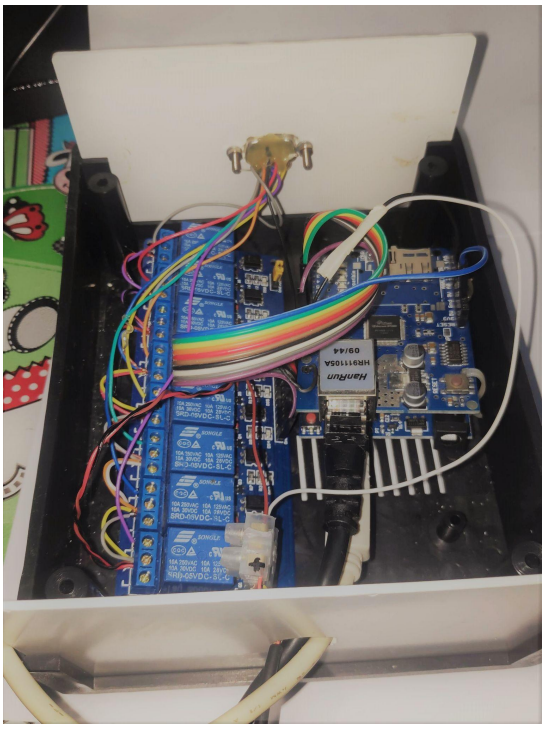


Figura 31: Módulo processador receptor - Esquerda placa com 8 relés para ativar as saídas e direita, processador Arduino com uma placa ethernet.

Para melhor apresentação dos níveis lógicos desta saída, comparando com que foi enviado pela interface de entrada, foi desenvolvido a interface de saída (Figura 32) que sinaliza por leds qual o status atual da última mensagem recebida.

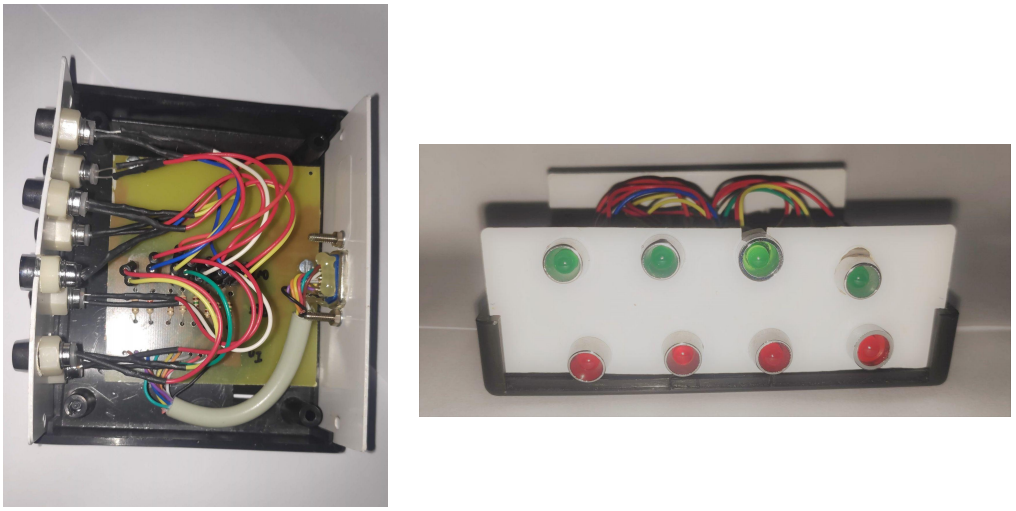


Figura 32: Módulo processador receptor - Esquerda foto do circuito montado e direita, painel de Leds para exibição dos dados,

4.1 Testes de funcionamento

Como procedimento de teste, primeiro enviou-se um bit de cada vez para o sistema, acionando-se chave por chave na interface de entrada. E como esperado, uma a uma, as portas de saída foram acionadas, ligando os leds da interface de saída.

Depois, o teste aplicado foi alterar a condição de várias entradas ao mesmo tempo, o que ocorre frequentemente no sistema ferroviário. E o sistema respondeu perfeitamente, alterando todas as saídas conforme acionadas.

O vídeo deste teste por ser visto no link:

https://www.youtube.com/watch?v=Dsx7YyQguxE

6. Conclusão

Com tudo construído e ligado, foi possível comprovar a funcionalidade do projeto. Os dados inseridos eletricamente do lado transmissor foram corretamente enviados pelo sistema óptico e foram convertidos novamente em sinais elétricos, conforme projetado.

O protótipo que foi construído foi fundamental para exibir os resultados esperados. Com sinais luminosos na entrada e saída, foi possível “ver” *bit* a *bit*, o que estava sendo transmitido e o que foi recebido.

Um ponto importante a ser considerado é utilizar o conversor de mídia correto para a aplicação. Devido a longa distância entre um ponto e outro de comunicação, atentar-se aos limites dos transmissores ópticos é fundamental. No caso deste protótipo foi utilizado um modelo com capacidade de até 25 km.

Porém, devido aos problemas causados pela pandemia, os testes foram realizados apenas com um cordão de fibra óptica de 1,5m. Seria necessário um laboratório com um longo cordão de fibra óptica e um medidor de potência para verificar a atenuação.

Analisando o projeto como um todo, é possível perceber que ele não só minimiza os efeitos dos furtos que existem na região, mas também geram uma economia enorme em relação ao custo da instalação do sistema. Cabos de cobre, que são utilizados no sistema atual, são consideravelmente mais caros do que a fibra óptica.

E da maneira como este projeto foi elaborado, podemos substituir 60 fios de cobre por apenas 1 cordão de fibra óptica. Pois, em um módulo processador é possível instalar 15 placas de entrada e cada placa comporta até 4 bits de informação.

Também diminui-se o tempo de resolução de falhas, o que neste caso de transporte público ferroviário é importantíssimo. Já que para realizar a fusão de um cordão de fibra óptica gasta-se muito menos tempo do que 60 emendas de fios de cobre.

Como proposta de um projeto complementar a este, para aumentar a robustez do sistema, poderia implantar um sistema de redundância por antenas. Quando detectado algum problema na transmissão óptica, comutaria automaticamente para as antenas, mantendo o sistema ativo.

6. Referências Bibliográficas

[1] ANTUNES, Camila; FERREIRA, Fernanda; PÁSSARO, Thiago. Entre Trilhos: Uma viagem de história pela primeira estrada de ferro paulista, a São Paulo Railway. 1. ed. São Bernardo do Campo: [s.n.], 2014. 145 p. v. 1.

[2] GIESBRECHT, RALPH MENNUCCI. Estações Ferroviárias do Brasil: Rio Grande da Serra. 2016. Disponível em: <http://www.estacoesferroviarias.com.br/r/rgserra.htm>. Acesso em: 17 ago. 2018.

[3] CPTM, Companhia Paulista de Trens Metropolitanos. Furtos de Cabos: Resumo. 2018. Disponível em: <https://extranet.cptm.sp.gov.br/Pages/default.aspx>. Acesso em: 17 ago. 2018.

[4] CPTM registra queda no número de passageiros na Linha 10. ABC: Repórter Diário, 2018. Disponível em: <https://www.reporterdiario.com.br/noticia/2523632/cptm-registra-queda-no-numero-de-passageiros-na-linha-10/>. Acesso em: 17 ago. 2018.

[5] HISTÓRIA da Ferrovia no Brasil. 2017. Disponível em: <http://www.brasilferroviario.com.br/historia-da-ferrovia-no-brasil/>. Acesso em: 17 ago. 2018.

[6] ESTADO DE SÃO PAULO. Lei n. 7.861, de 28 de maio de 1992. Autoriza o Poder Executivo a constituir a Companhia Paulista de Trens Metropolitanos - CPTM. , e dá outras providências. LUIZ ANTONIO FLEURY FILHO. Palácio dos Bandeirantes, p. 1-1, maio. 1992. Disponível em: <https://www.al.sp.gov.br/repositorio/legislacao/lei/1992/lei-7861-28.05.1992.html>. Acesso em: 17 ago. 2018.

[7] SILVA, Júlio César Lázaro da. "Breve História das Ferrovias"; Brasil Escola. Disponível em <https://brasilescola.uol.com.br/geografia/ferrovias.htm>. Acesso em 17 de agosto de 2018.

[8] KUROSE, James F.; ROSS, Keith W. Redes de Computadores e a Internet: uma abordagem top-down. 6. ed. New Jersey: Pearson, 2015. 656 p. ISBN 978-0-13-285620-1.

7. Anexos

7.1 Código Arduino Transmissor

```
#include <SPI.h>
#include <Ethernet.h>
#include <EthernetUdp.h>
#include <math.h>

// Configuracao do endereço MAC e IP do controlador.
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress ip(192, 168, 1, 177);

//Dados do endereco de destino.
byte server_ip[]        = { 192,168,1,178 };
uint16_t targetPort     = 8887;

unsigned int localPort = 8888;//porta local para comunicacao.

EthernetUDP Udp;

bool G[4][7] = { //Matriz geradora para calculo do codigo de Hamming

  {true, true, false, true, false, false, false },

  {false, true, true, false, true, false, false },

  {true, true, true, false, false, true, false },

  {true, false, true, false, false, false, true }

};

 void enviar(unsigned char mensagem){

  Udp.beginPacket(server_ip, targetPort);
  Udp.write(mensagem);
  Udp.endPacket();
  }

void setup() {

  //Iniciando Ethernet e UDP.
  Ethernet.begin(mac,ip);
  Udp.begin(localPort);

  //Configurando porta de comunicacao serial
  Serial.begin(9600);
```

```
  //Configurando pinos de entrada de dados do controlador.
  pinMode(31, INPUT);
  pinMode(32, INPUT);
  pinMode(33, INPUT);
  pinMode(34, INPUT);
  pinMode(35, INPUT);
  pinMode(36, INPUT);
  pinMode(37, INPUT);
  pinMode(38, INPUT);
 }

 void loop() {

   bool x[7] = {false, false, false, false, false, false, false}; // Vetor que forma byte de saída
  bool in1, in2, in3, in4; //Variaveis de id das placas

  //Montando pacote de enderecamento da placa
  unsigned char inf = 0; //Flag do tipo da mensagem: 0 = endereco
  unsigned char send = 0; //Variavel com o valor a ser transmitido

  //Placa 1d = 0001b
  in1 = false;
  in2 = false;
  in3 = false;
  in4 = true;

  bool vet1[4] = {in1, in2, in3, in4};
  for (int j = 0; j<7; j++){
    for (int i = 0; i<4; i++){
      bool aux = vet1[i] & G[i][j];
      x[j] = x[j] ^ aux;
    }
    send = send + (x[j] << (7-j));
  }
  send = send + inf;
  Serial.println(send, BIN);
  enviar (send);
  delay(500);

  //Montando pacote de dados
  send = 0;
  for (int i=0;i<7;i++) x[i] = false;
  inf = 1; // Mensagem do tipo informacao

   // Leitura de 4 entradas placa 1
  if (digitalRead(34)==HIGH) in1 = true;
  else in1 = false;
```

```
if (digitalRead(33)==HIGH) in2 = true;
else in2 = false;
if (digitalRead(32)==HIGH) in3 = true;
else in3 = false;
if (digitalRead(31)==HIGH) in4 = true;
else in4 = false;

vet1[0] = in1;
vet1[1] = in2;
vet1[2] = in3;
vet1[3] = in4;

for (int j = 0; j<7; j++){
  for (int i = 0; i<4; i++){
    bool aux = vet1[i] & G[i][j];
    x[j] = x[j] ^ aux;
  }
  send = send + (x[j] << (7-j));
}
send = send + inf;
Serial.println(send, BIN);
enviar (send);
delay(500);

//Montando pacote de enderecamento da placa
//Placa 2 = 0010
in1 = false;
in2 = false;
in3 = true;
in4 = false;

vet1[0] = in1;
vet1[1] = in2;
vet1[2] = in3;
vet1[3] = in4;

inf = 0;
for (int i=0;i<7;i++) x[i] = false;
send = 0;

for (int j = 0; j<7; j++){
  for (int i = 0; i<4; i++){
    bool aux = vet1[i] & G[i][j];
    x[j] = x[j] ^ aux;
  }
  send = send + (x[j] << (7-j));
}
send = send + inf;
Serial.println(send, BIN);
```

```
  enviar (send);
  delay(500);

  //// Leitura de 4 entradas placa 2
  send = 0;
  for (int i=0;i<7;i++) x[i] = false;
  inf = 1;

  if (digitalRead(38)==HIGH) in1 = true;
  else in1 = false;
  if (digitalRead(37)==HIGH) in2 = true;
  else in2 = false;
  if (digitalRead(36)==HIGH) in3 = true;
  else in3 = false;
  if (digitalRead(35)==HIGH) in4 = true;
  else in4 = false;

  vet1[0] = in1;
  vet1[1] = in2;
  vet1[2] = in3;
  vet1[3] = in4;

  for (int j = 0; j<7; j++){
    for (int i = 0; i<4; i++){
      bool aux = vet1[i] & G[i][j];
      x[j] = x[j] ^ aux;
    }
    send = send + (x[j] << (7-j));
  }
  send = send + inf;
  Serial.println(send, BIN);
  enviar (send);
  delay(500);
}
```

7.2 Código Arduíno Receptor

```
#include <SPI.h>
#include <Ethernet.h>
#include <EthernetUdp.h>

// Dados do endereço MAC e IP do controlador.
byte mac[] = {
  0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xEE };
IPAddress ip(192, 168, 1, 178);

unsigned int localPort = 8887;  //Porta local

EthernetUDP Udp;

char packetBuffer[UDP_TX_PACKET_MAX_SIZE]; //buffer para armazenar pacotes recebidos

bool Ht[7][3] = { //Matriz para calculo da sindrome

  {true, false, false },

  {false, true, false },

  {false, false, true },

  {true, true, false },

  {false, true, true },

  {true, true, true },

  {true, false, true }

};

int placa = 0;

void setup() {
  //Inicio dos protocolos Ethernet e UDP
  Ethernet.begin(mac,ip);
  Udp.begin(localPort);
  Serial.begin(9600);

  //Inicializando saidas do controlador
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
```

```
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
 }

 void loop() {

   int packetSize = Udp.parsePacket(); //Recebendo os dados do buffer de
entrada
  Serial.print("Received packet of size ");
   Serial.println(packetSize);
   Serial.print("From ");
   IPAddress remote = Udp.remoteIP();
   for (int i =0; i < 4; i++)
   {
    Serial.print(remote[i], DEC);
    if (i < 3)
    {
     Serial.print(".");
    }
   }
   Serial.print(", port ");
   Serial.println(Udp.remotePort());

   //Leitura da informacao dentro do buffer
   Udp.read(packetBuffer,1);
   Serial.println("Contents:");
   Serial.println((unsigned char)packetBuffer[0], BIN);
   int receive = (unsigned char)packetBuffer[0];

  if(packetSize)
   {
   bool r[7] = {false, false, false, false, false, false, false};
   bool sindrome[3] = {false, false, false};

   int k, mod;
   int sind = 0;
   byte integridade = 1;

   //Fator k determina quantos bits tem a mensagem
   if (receive>=128) k = 0;
   else if (receive>=64) k = 1;
   else if (receive>=32) k = 2;
   else if (receive>=16) k = 3;
   else if (receive>=8) k = 4;
   else if (receive>=4) k = 5;
```

```
else if (receive>=2) k = 6;
else if (receive<2) k = 7;

if (receive%2 == 1) mod = 1;
else mod = 0;
Serial.print("Mod: ");
Serial.println(mod);

for (int i=7;i>=k;i--){
  if (receive%2==0) r[i] = false;
  else r[i] = true;
  receive = receive/2;
}

for (int j = 0; j<3; j++){
  for (int i = 0; i<7; i++){
    bool aux = r[i] & Ht[i][j];
    sindrome[j] = sindrome[j] ^ aux;
  }
  sind = sind + (sindrome[j] << (2-j));
}

  Serial.print("sindrome : ");
  Serial.println(sind);

switch (sind) {
  case 1:
    r[2] = !r[2];
    break;
  case 2:
    r[1] = !r[1];
    break;
  case 3:
    r[4] = !r[4];
    break;
  case 4:
    r[0] = !r[0];
    break;
  case 5:
    r[6] = !r[6];
    break;
  case 6:
    r[3] = !r[3];
    break;
  case 7:
    r[5] = !r[5];
    break;
  case 0:
    integridade = 0;
```

```
    break;
}

if (integridade != 0){
  //Zerando sindrome
  for (int i=0; i<3; i++) sindrome[i]=false;
  sind = 0;

  //Recalculando sindrome
  for (int j = 0; j<3; j++){
    for (int i = 0; i<7; i++){
      bool aux = r[i] & Ht[i][j];
      sindrome[j] = sindrome[j] ^ aux;
    }
    sind = sind + (sindrome[j] << (2-j));
  }
  Serial.print("sindrome 2: ");
  Serial.println(sind);
  if (sind==0) integridade = 0;
}

if (integridade == 0){
  if (mod==0){ //recebendo dados de enderecamento
    placa = 0;
    if (r[3]==true) placa = placa + 8;
    if (r[4]==true) placa = placa + 4;
    if (r[5]==true) placa = placa + 2;
    if (r[6]==true) placa = placa + 1;
    Serial.print("placa: ");
    Serial.println(placa);
  }

  else if (mod==1 && placa!=0){ //recebendo dados de saida
      if (r[3]==true) {
        digitalWrite((2+(placa-1)*4), HIGH);
        Serial.println("Out 2: on");
      }
      else digitalWrite((2+(placa-1)*4), LOW);
      if (r[4]==true) {
        digitalWrite((3+(placa-1)*4),HIGH);
        Serial.println("Out 3: on");
      }
      else digitalWrite((3+(placa-1)*4), LOW);
      if (r[5]==true) {
        digitalWrite((4+(placa-1)*4),HIGH);
        Serial.println("Out 4: on");
      }
      else digitalWrite((4+(placa-1)*4), LOW);
      if (r[6]==true) {
```

```
          digitalWrite((5+(placa-1)*4),HIGH);
          Serial.println("Out 5: on");
        }
        else digitalWrite((5+(placa-1)*4), LOW);
      }
    }
  }
}
```

7.3 Manual Conversor de Mídia Planet

1. Características do Produto

FT-80x Fast Ethernet Conversor de Mídia:

- De acordo com IEEE 802.3/802.3u, 10/100Base-TX, 100Base-FX Standard
- Taxa de Transmissão de Dados: TP - 10/100Mbps; FX - 100Mbps
- Modo de suporte Duplex:
  - Modo inteiro ou meio-duplex por Auto-Negociação (Porta Twisted Pair)
  - Modo inteiro ou meio-duplex por interruptor DIP- (Porta de Fibra)
- Interruptor DIP: 2 interruptores DIP; modo de seleção FX duplex & modo de seleção LFP
- De acordo com o Chassis de Conversão de Mídia PLANET 10"/19"

GT-80x Gigabit Ethernet Conversor de Mídia:

- De acordo com IEEE 802.3 10Base-T, IEEE 802.3u 100Base-TX, IEEE 802.3ab 1000Base-T, IEEE 802.3z 1000Base-SX/LX standard
- Taxa de Transmissão de Dados: TP – 10/100/1000Mbps; FX – 1000Mbps
- Interruptor DIP para configuração de função (Desabilitado/Habilitado) LFP
- OAM (TS-1000 and IEEE 802.3ah) suportado
- Quadro 9K Jumbo suportado
- De acordo com o Chassis de Conversão de Mídia PLANET 10"/19"

Nas seções seguintes, o termo **"FT-80x / GT-80x"** indica a família de produtos acima, o termo **"MM"** e **"SM"** representa Módulo Múltiplo e Módulo Singular de fibra-óptica.

2. Checklist

A sua caixa com FT-80x / GT-80x deve conter os seguintes itens:

- O Conversor de Mídia Fast / Gigabit Ethernet
- Adaptador de Energia AC-DC (Saída: 5V DC, 2A máx.)
- Este Manual do Usuário

Se qualquer item esteja faltando ou esteja danificado, por favor consulte o revendedor de quem você comprou o seu Conversor de Mídia Fast / Gigabit Ethernet.

Nota	GT-805A está com um espaço de módulo SFP vago. O módulo mini GBIC SFP não vem em conjunto com este pacote.

3. Panorama do Produto

Visualização Direita (FT-80x)

Existem uma tomada RJ-45 Twisted-Pair (Auto-MDI/MDI-X), um conector de fibra-óptica (varia de acordo com o modelo) e seis indicadores de LED.

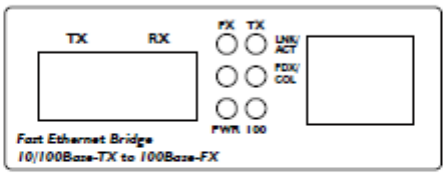


Figura 1: FT-80x Visualização Direita

Visualização Direita (GT-80x)

Existem uma tomada RJ-45 Twisted-Pair (Auto-MDI/MDI-X), um conector de fibra-óptica (varia de acordo com o modelo) e quatro indicadores de LED. Também um interruptor DIP para atributo de Link-Fault -Ppassthrough (LFP), "ON" para ligar o LLCF e detectar o LLR. E "OFF" para desligar o atributo. Por favor refira às seções seguintes para mais.

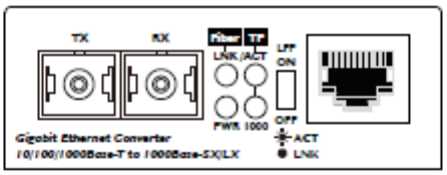


Figura 2: GT-802/802S Visualização Direita

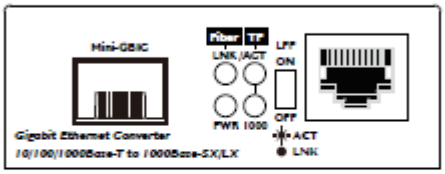


Figura 3: GT-805A Visualização Direita

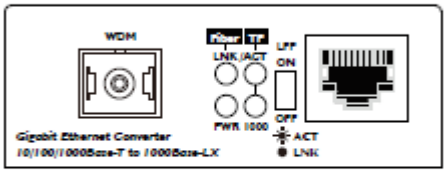
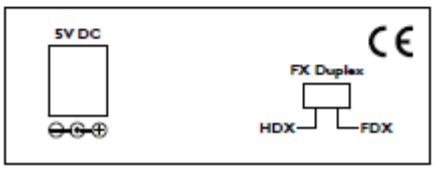


Figura 4: GT-806A15/B15/A60/B60 Visualização Direita

Visualização Esquerda (FT-80x)

Um soquete de energia DC 5V e um interruptor DIP para operação do modo de seleção de fibra-óptica, FDX para duplex inteiro, e HDX para meio-duplex.



Visualização Esquerda (GT-80x)

O painel de trás do Conversor de Mídia Gigabit Ethernet indica um soquete DC, que aceita entrada de energia com 5V DC 2A..

Figura 6: GT-80x Visualização

Visualização Lateral (FT-80x)

Um interruptor DIP para o atributo Link Fault Pass Through (LFP), "ON" para ligar o LLCF e detectar LLR. E "OFF" para desligar este atributo. Por favor refira às seções seguintes para mais.

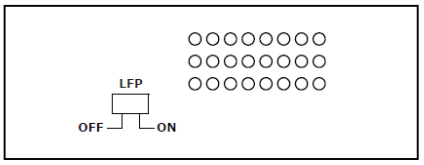


Figura 7: FT-80x Visualização Lateral

4. Link Fault Pass Through (LFP)

A função LFP inclui a função Link-Fault-Passthrough (LLCF/LLR) e o design de Interruptor DIP. LLCF/LLR podem imediatamente alarmar os administradores a respeito do problema de mídia com um link e providenciar solução eficiente para monitorar a rede. O Interruptor DIP irá desativar ou ativar a função LFP.

LLCF (Link Loss Carry Forward) significa quando o dispositivo conectado ao conversor e o TP link de perda de linha, o fibra do conversor irá disconectar o link de transmissão. LLR (Link Loss Return) significa quando o dispositivo conectado ao conversor e o link de fibra de perda de linha, a fibra do conversor irá disconectar o link de transmissão.


Nota

A função LFP está ligada de acordo com a configuração padrão. Se você é familiar com a intalação de rede e a propósito de diagnóstico (por ex., checar qual final está quebrado), você pode desligar e restabelecer o conversor para fazer com que surta efeito. Caso contrário, por favor deixe na posição padrão.

5. Instalando o Conversor

Para instalar o FT-80x / GT-80x independentement, em uma área de trabalho ou prateleira, simplesmente complete os passos seguintes:

Passo 1: Desligue a energia do dispositivo / estação na rede na qual o FT-80x / GT-80x será anexado.

Passo 2: Ligue o cabo de fibra do FT-80x / GT-80x à fibra da rede. **TX**, **RX** precisam ser emparceirados em ambos os finais.

Passo 3: Ligue um cabo Cat. 5/5e/6 UTP da rede 10/100Base-TX ou 10/100/1000Base-T para a porta RJ-45 no FT-80x / GT-80x.

Passo 4: Conecte o adaptador de energia 5VDC ao FT-80x / GT-80x e verifique se Power LED acende.

Passo 5: Ligue a energia do dispositivo / estação, os TX Link e FX Link LEDs devem acender quando todos os cabos estiverem conectados.

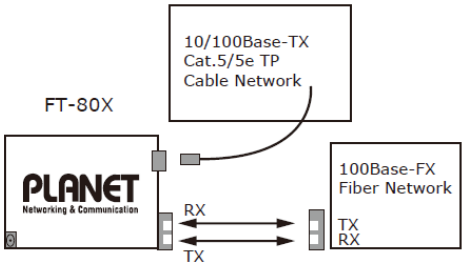


Figura 8: Instalação do FT-80x

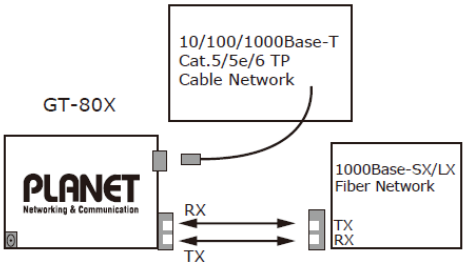


Figura 9: Instalação do GT-80x

Nota

1. É recomendada a utilização de PLANET MGB-SX / MGB-LX séries 1000Base-SX/LX SFP no GT-805A. Se você inserir um transceptor SFP que não é suportado, o GT-805A não o reconhecerá.
2. Por favor cheque o link-budget de seu transceptores SFP e suas distacias físicas de fiação, em uma instalação, um atenuador em-linha pode ser requerido para projetar os transceptores.

6 Modo Duplex de Suporte

A porta FT-80x TP suporta o modo duplex de detecção por auto-negociação (A-N). Isto deve detectar automaticamente a velocidade do link e o modo duplex padrão. E o fall-back automático de 100 ou 10Mbps modo meio-duplex com o seu link parceiro. A porta de fibra suporta a seleção de interruptor DIP para Duplex Inteiro (padrão) ou Meio-duplex.

A porta GT-80x TP suporta tripla velocidade 10/100/1000Base-T, Auto-negociação. Isto deve detectar automaticamente a velocidade do link e modo duplex padrão com o seu link parceiro. A porta de Fibra 1000Base-SX/LX, permite 1000Mbps duplex inteiro por Auto-Negociação. Por favor também cheque a configuração do link parceiro.

7. Indicador LED

FT-80x:

LED	Cor	Descrição
FX LNK / ACT	Verde	**Pisca:** quando qualquer pacote FX transmitindo e recebendo.
		Ilumina: quando a conexão de Fibra é boa.
TX LNK / ACT	Verde	**Pisca:** quando qualquer pacote TP transmitindo e recebendo.
		Ilumina: quando a conexão de Fibra é boa.
FX FDX / COL	Verde	**Ilumina:** quando o modo Duplex inteiro está ativado na porta FX.
		Pisca: quando a porta FX está em modo meio-duplex e recebe colisão.
TX FDX / COL	Verde	**Ilumina:** quando o modo Duplex inteiro está ativado (detectado pela Auto-Negociação) na porta TP.
		Pisca: quando a porta TP recebe a colisão.
100	Verde	**Ilumina:** quando a porta TP funciona com 100Mbps. Continua desligada enquanto a iluminação LINK LED representa a porta TP funcionando em 10Mbps.
PWR	Verde	**Ilumina:** quando a energia +5VDC é detectada.

8 Parâmetro de Conexão do Cabo

Cabos:

Padrão (Cumprimento de Onda)	100Base-FX (1310nm)	1000Base-SX (850nm)	1000Base-LX (1310nm)
Tipo de Fibra & Especificação do Cabo	Modo Múltiplo	50/125µm or 62.5/125µm	
	Modo Singular	9/125µm	

FT-80x:

Os detalhes da fiação estão como abaixo:
Distância da fiação:

Duplex	Conexão	Limite (máx.)
Twisted Pair		
Meio / Inteiro	De Nó a Nó De Nó a Interruptor/Eixo	100 metros
Conversores Modo Múltiplo		
MM Meio	De Nó a Nó De Nó a Interruptor	412 metros
MM Inteiro	De Nó a Nó De Nó a Interruptor	2 quilometros
Conversores de Modo Singular* (FT-80xynn; x= 2, 6; y= S, A, B; nn=km)		
SM Inteiro	De Nó a Nó De Nó a Interruptor	Depende do modelo

9. Especificação do Produto

FT-80x

Modelo	FT-801	FT-802	FT-802S15	FT-802S35	FT-802S50	FT-806A20 FT-806B20
Conector de Fibra	ST	SC	SC			SC WDM
Modo de Fibra	Modo Múltiplo		Modo Singular			
Distância Máxima da Fibra	2km		15km	35km	50km	20km
Conector de Cobre	10/100Mbps RJ-45					
Modos de Cobre	Duplex Inteiro, auto-negociação					
Taxa de Encaminhamento de Pacote (64bytes)	14880pps @10Mbps; 148810pps @100Mbps					
Interruptor DIP	LFP Desativado / Ativado; FX Inteiro / Meio-duplex					
Protocolos e Padrões	IEEE 802.3, 10Base-T IEEE 802.3u, 100Base-TX, 100Base-FX					
Dimensões	97 x 69 x 26 mm (L x P x A)					
Peso	0.2kg					
Energia	5VDC, 2A máx.					
Emissão	FCC, CE					
Temperatura	Operação: 0 ~ 50℃ / Armazenagem: -40 ~ 70℃					
Umidade	5% ~ 95% sem condensação					
Instalação	Montagem na parede, trilho DIN, Instalação de Chassis					

Nota

1. Por favor note que FT-806A20/806B20 é desenhado para trabalhar junto. Significa que você precisa conectar o FT-806A20 ao FT-806B20 em par. Se ambos os finais são FT-806A20 (or FT-806B20), ou qualquer modelo acima a qualquer dispositivo de parte terceira, eles não podem trabalhar normalmente e podem danificar os conectores de fibras.
2. A série FT-802xxx / FT-806xxx de Conversores de Mídia de modo singular fornecem suporte de longa distância de15km a 50km. Quando a distância do cabo de fibra de modo singular for menor, você pode precisar inserir um atenuador óptico em linha ao link para prevenir a sobrecarga do receptor:
3. Ao se conectar aos produtos do Fast Ethernet, por favor refira ao Manual Técnico do dispositivo.
4. Consulte o seu revendedor para trilho DIN ou instalação de Chassis.

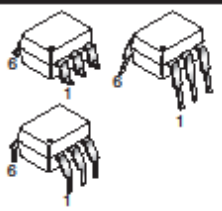
7.4 Datasheet Optoacoplador 4N25

FAIRCHILD
SEMICONDUCTOR®

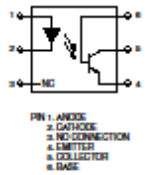
GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

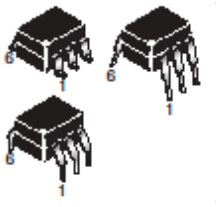
WHITE PACKAGE (-M SUFFIX)



SCHEMATIC



BLACK PACKAGE (NO -M SUFFIX)



DESCRIPTION

The general purpose optocouplers consist of a gallium arsenide infrared emitting diode driving a silicon phototransistor in a 6-pin dual in-line package.

FEATURES

- Also available in white package by specifying -M suffix, eg. 4N25-M
- UL recognized (File # E90700)
- VDE recognized (File # 94766)
  - Add option V for white package (e.g., 4N25V-M)
  - Add option 300 for black package (e.g., 4N25.300)

APPLICATIONS

- Power supply regulators
- Digital logic inputs
- Microprocessor inputs

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C unless otherwise specified)

Parameter	Symbol	Value	Units
TOTAL DEVICE			
Storage Temperature	T_{STG}	-55 to +150	°C
Operating Temperature	T_{OPR}	-55 to +100	°C
Wave solder temperature (see page 14 for reflow solder profiles)	T_{SOL}	260 for 10 sec	°C
Total Device Power Dissipation @ T_A = 25°C	P_D	250	mW
Derate above 25°C		3.3 (non-M), 2.94 (-M)	
EMITTER			
DC/Average Forward Input Current	I_F	100 (non-M), 60 (-M)	mA
Reverse Input Voltage	V_R	6	V
Forward Current - Peak (300µs, 2% Duty Cycle)	I_F(pk)	3	A
LED Power Dissipation @ T_A = 25°C	P_D	150 (non-M), 120 (-M)	mW
Derate above 25°C		2.0 (non-M), 1.41 (-M)	mW/°C
DETECTOR			
Collector-Emitter Voltage	V_{CEO}	30	V
Collector-Base Voltage	V_{CBO}	70	V
Emitter-Collector Voltage	V_{ECO}	7	V
Detector Power Dissipation @ T_A = 25°C	P_D	150	mW
Derate above 25°C		2.0 (non-M), 1.76 (-M)	mW/°C

ELECTRICAL CHARACTERISTICS (T_A = 25°C unless otherwise specified)

INDIVIDUAL COMPONENT CHARACTERISTICS

Parameter	Test Conditions	Symbol	Min	Typ*	Max	Unit
EMITTER						
Input Forward Voltage	(I_F = 10 mA)	V_F		1.18	1.50	V
Reverse Leakage Current	(V_R = 6.0 V)	I_R		0.001	10	µA
DETECTOR						
Collector-Emitter Breakdown Voltage	(I_C = 1.0 mA, I_F = 0)	BV_{CEO}	30	100		V
Collector-Base Breakdown Voltage	(I_C = 100 µA, I_F = 0)	BV_{CBO}	70	120		V
Emitter-Collector Breakdown Voltage	(I_E = 100 µA, I_F = 0)	BV_{ECO}	7	10		V
Collector-Emitter Dark Current	(V_{CE} = 10 V, I_F = 0)	I_{CEO}		1	50	nA
Collector-Base Dark Current	(V_{CB} = 10 V)	I_{CBO}			20	nA
Capacitance	(V_{CE} = 0 V, f = 1 MHz)	C_{CE}		8		pF

ISOLATION CHARACTERISTICS

Characteristic	Test Conditions	Symbol	Min	Typ*	Max	Units
Input-Output Isolation Voltage	(Non '-M', Black Package) (f = 60 Hz, t = 1 min)	V_{ISO}	5300			Vac(rms)
	('-M', White Package) (f = 60 Hz, t = 1 sec)		7500			Vac(pk)
Isolation Resistance	(V_{I-O} = 500 VDC)	R_{ISO}	10^{11}			Ω
Isolation Capacitance	(V_{I-O} = &, f = 1 MHz)	C_{ISO}		0.5		pF
	('-M' White Package)			0.2	2	pF

Note
* Typical values at T_A = 25°C

TRANSFER CHARACTERISTICS (T_A = 25°C Unless otherwise specified.)

DC Characteristic	Test Conditions	Symbol	Device	Min	Typ*	Max	Unit
Current Transfer Ratio, Collector to Emitter	(I_F = 10 mA, V_{CE} = 10 V)	CTR	4N35 4N36 4N37	100			%
			H11A1	50			
			H11A5	30			
			4N25 4N26 H11A2 H11A3	20			
			4N27 4N28 H11A4	10			
	(I_F = 10 mA, V_{CE} = 10 V, T_A = -55°C)		4N35 4N36 4N37	40			
	(I_F = 10 mA, V_{CE} = 10 V, T_A = +100°C)		4N35 4N36 4N37	40			
Collector-Emitter Saturation Voltage	(I_C = 2 mA, I_F = 50 mA)	$V_{CE\ (SAT)}$	4N25 4N26 4N27 4N28			0.5	V
	(I_C = 0.5 mA, I_F = 10 mA)		4N35 4N36 4N37			0.3	
			H11A1 H11A2 H11A3 H11A4 H11A5			0.4	
AC Characteristic							
Non-Saturated Turn-on Time	(I_F = 10 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{ON}	4N25 4N26 4N27 4N28 H11A1 H11A2 H11A3 H11A4 H11A5		2		µs
Non Saturated Turn-on Time	(I_C = 2 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{ON}	4N35 4N36 4N37		2	10	µs

TRANSFER CHARACTERISTICS (T_A = 25°C Unless otherwise specified.) (Continued)

AC Characteristic	Test Conditions	Symbol	Device	Min	Typ*	Max	Unit
Turn-off Time	(I_F = 10 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{OFF}	4N25 4N26 4N27 4N28 H11A1 H11A2 H11A3 H11A4 H11A5		2		µs
	(I_C = 2 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)		4N35 4N36 4N37		2	10	

* Typical values at T_A = 25°C

TYPICAL PERFORMANCE CURVES

Fig. 1 LED Forward Voltage vs. Forward Current (Black Package)

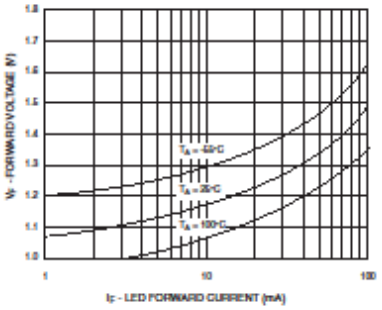


Fig. 2 LED Forward Voltage vs. Forward Current (White Package)

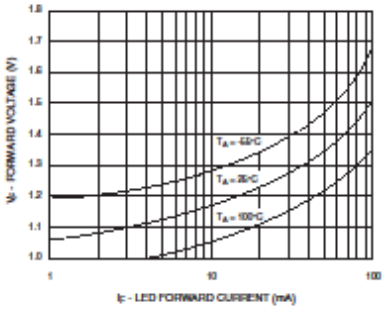


Fig.3 Normalized CTR vs. Forward Current (Black Package)

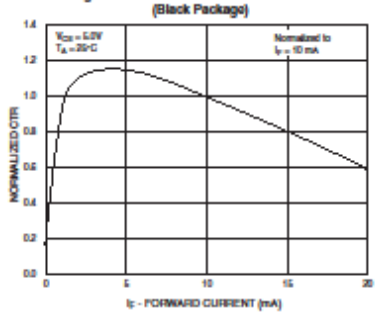


Fig.4 Normalized CTR vs. Forward Current (White Package)

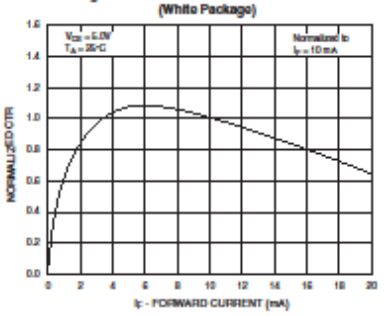


Fig. 5 Normalized CTR vs. Ambient Temperature (Black Package)

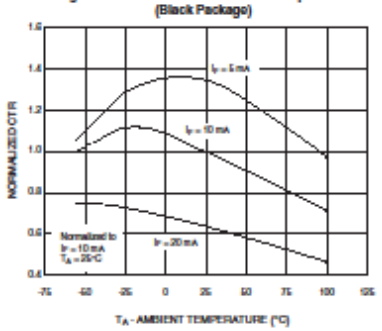


Fig. 6 Normalized CTR vs. Ambient Temperature (White Package)

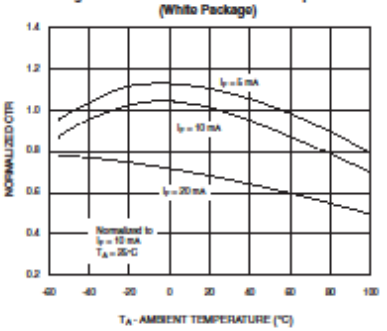


Fig. 7 CTR vs. RBE (Unsaturated) (Black Package)

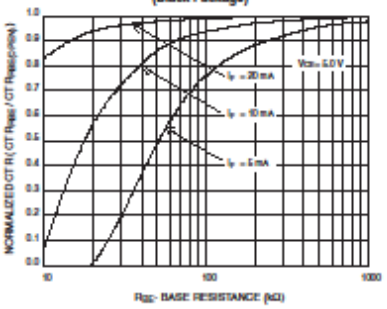


Fig. 8 CTR vs. RBE (Unsaturated) (White Package)

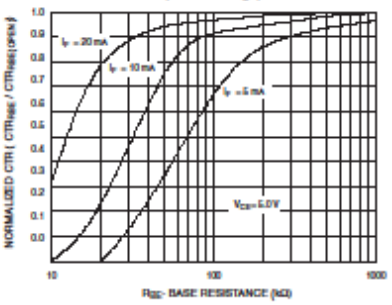


Fig. 9 CTR vs. RBE (Saturated) (Black Package)

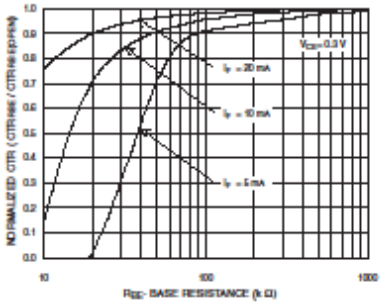


Fig. 10 CTR vs. RBE (Saturated) (White Package)

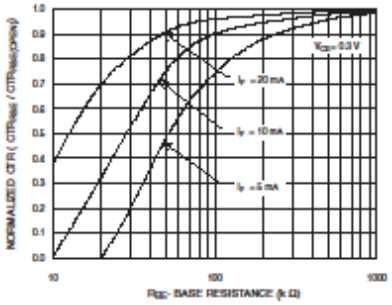


Fig. 11 Collector-Emitter Saturation Voltage vs Collector Current (Black Package)

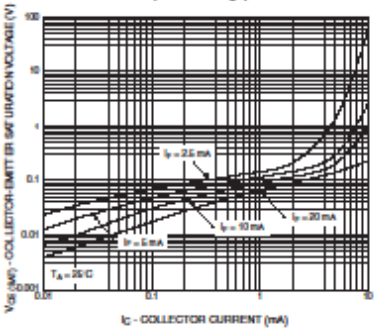


Fig. 12 Collector-Emitter Saturation Voltage vs Collector Current (White Package)

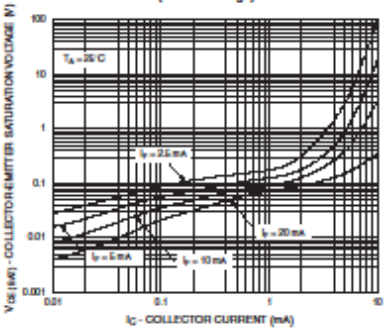


Fig. 13 Switching Speed vs. Load Resistor (Black Package)

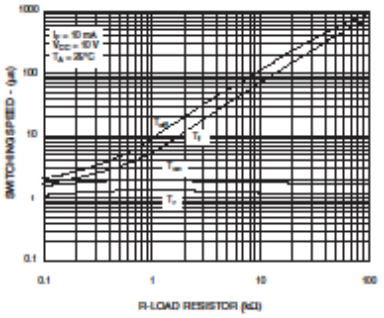


Fig. 14 Switching Speed vs. Load Resistor (White Package)

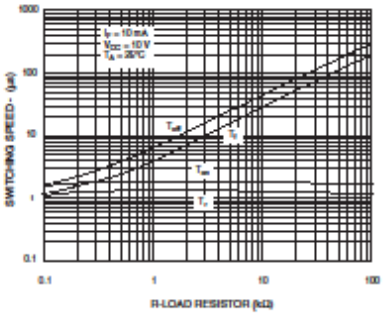


Fig. 15 Normalized t_{on} vs. R_{BE} (Black Package)

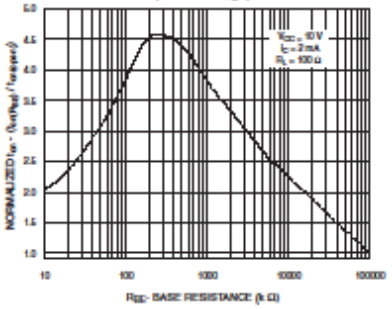


Fig. 16 Normalized t_{on} vs. R_{BE} (White Package)

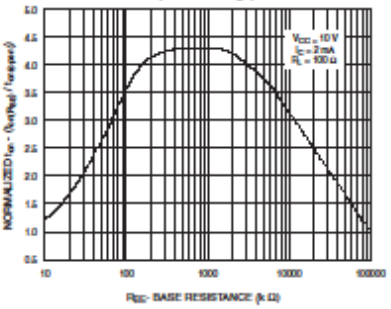


Fig. 17 Normalized t_{off} vs. R_{BE} (Black Package)

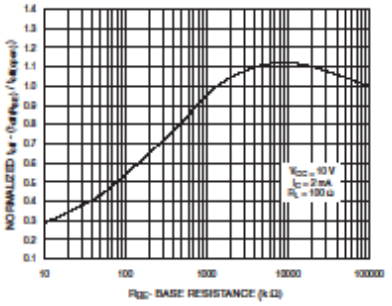


Fig. 18 Normalized t_{off} vs. R_{BE} (White Package)

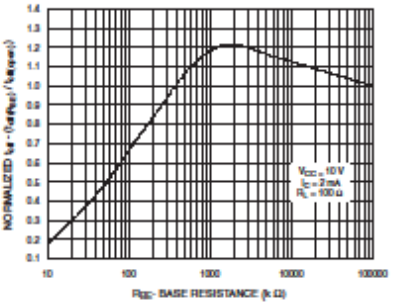
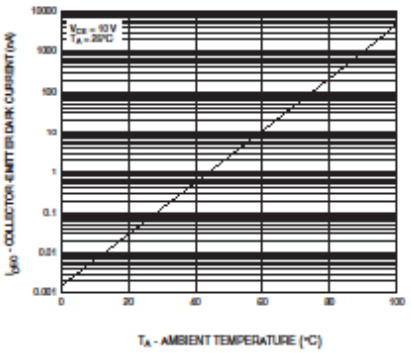


Fig. 19 Dark Current vs. Ambient Temperature



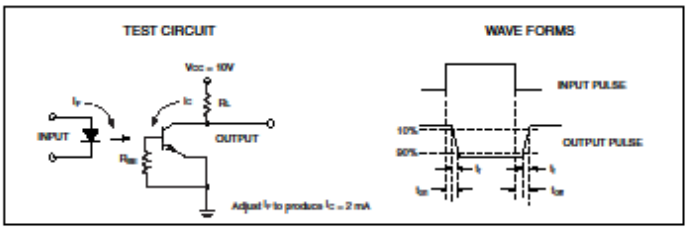
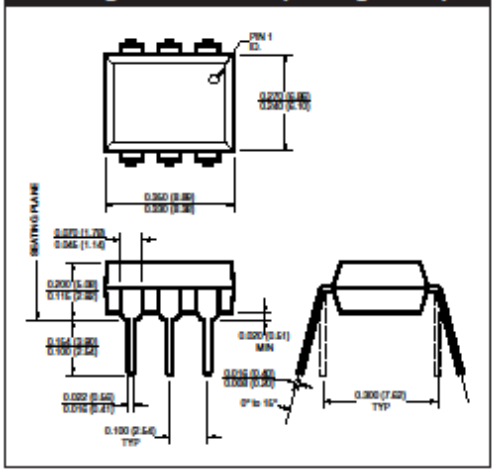


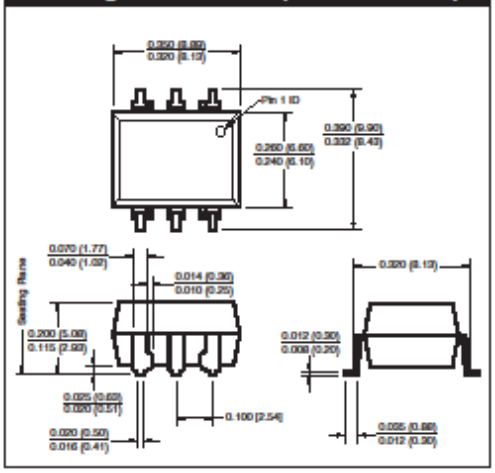
Figure 20. Switching Time Test Circuit and Waveforms

Black Package (No -M Suffix)

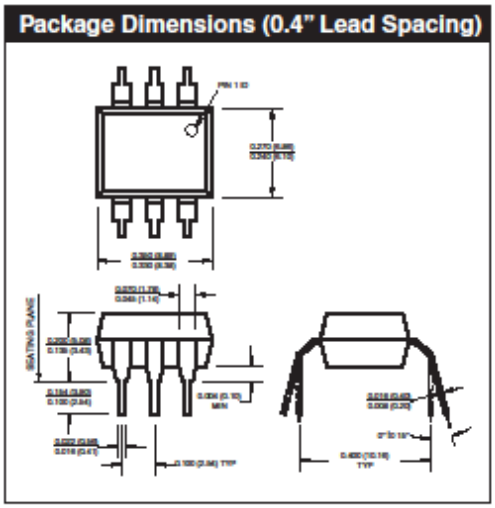
Package Dimensions (Through Hole)



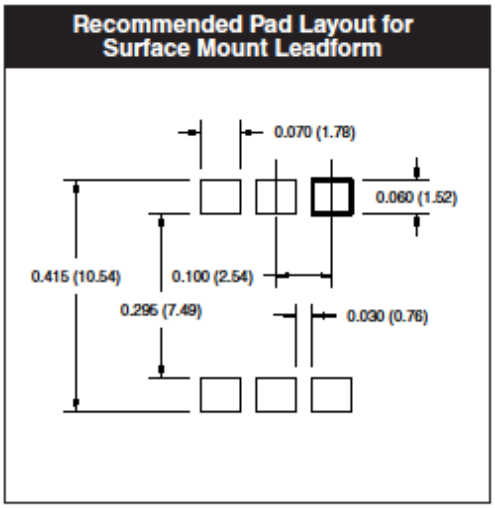
Package Dimensions (Surface Mount)



Package Dimensions (0.4" Lead Spacing)



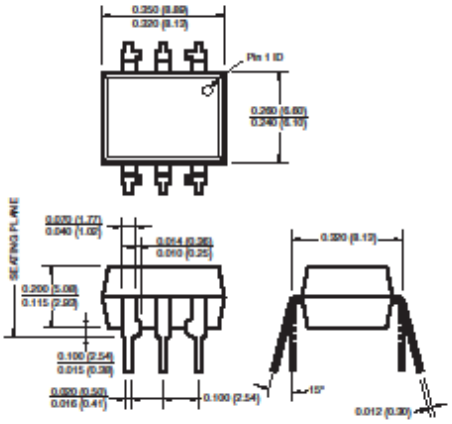
Recommended Pad Layout for Surface Mount Leadform



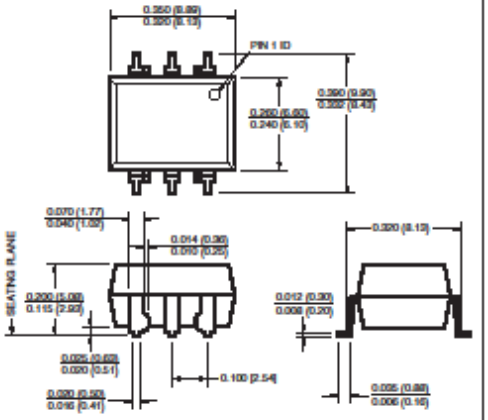
NOTE
All dimensions are in inches (millimeters)

White Package (-M Suffix)

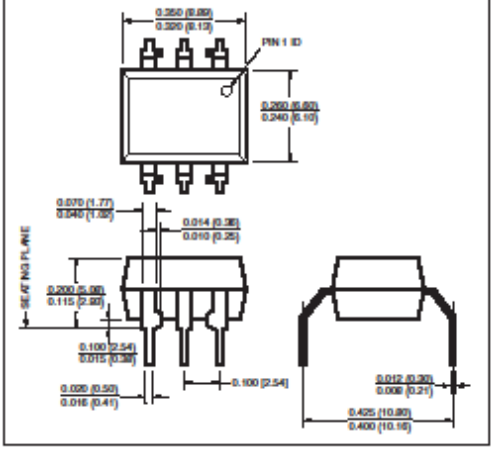
Package Dimensions (Through Hole)



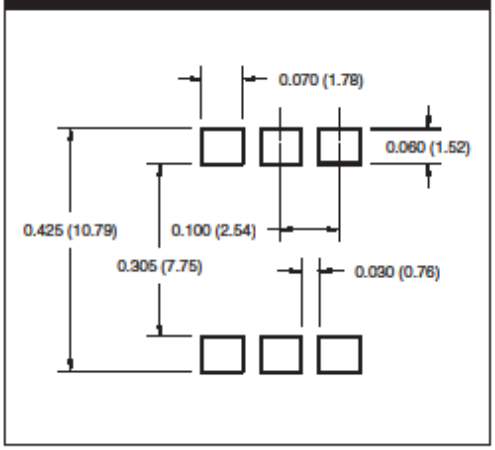
Package Dimensions (Surface Mount)



Package Dimensions (0.4" Lead Spacing)



Recommended Pad Layout for Surface Mount Leadform

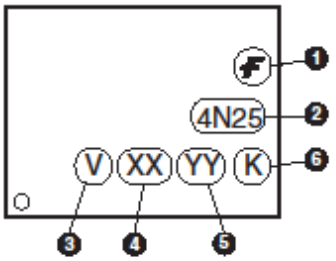


NOTE
All dimensions are in inches (millimeters)

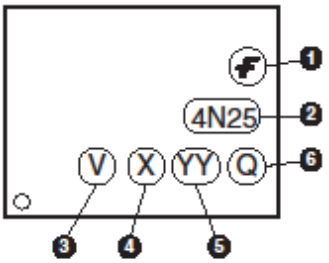
ORDERING INFORMATION

Order Entry Identifier		
Black Package (No Suffix)	**White Package (-M Suffix)**	**Option**
.S	S	Surface Mount Lead Bend
.SD	SR2	Surface Mount; Tape and reel
.W	T	0.4" Lead Spacing
.300	V	VDE 0884
.300W	TV	VDE 0884, 0.4" Lead Spacing
.3S	SV	VDE 0884, Surface Mount
.3SD	SR2V	VDE 0884, Surface Mount, Tape & Reel

MARKING INFORMATION



Black Package, No Suffix

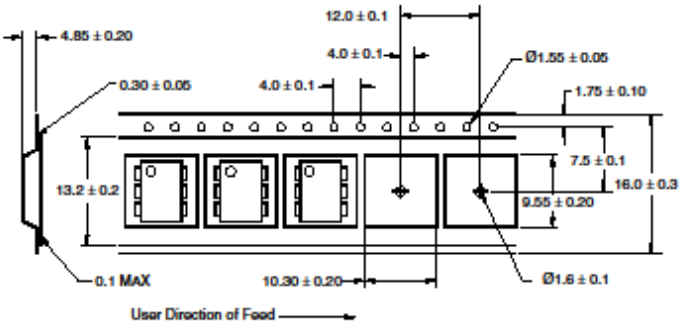


White Package, -M Suffix

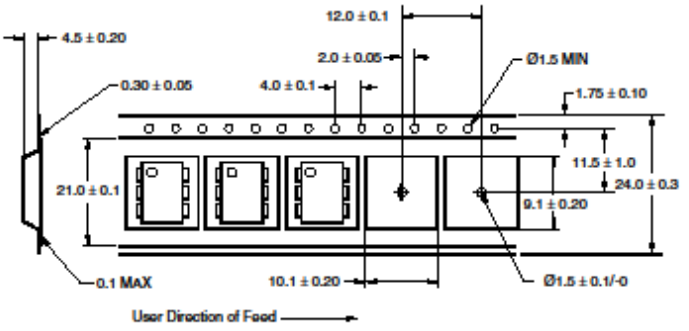
Definitions	
1	Fairchild logo
2	Device number
3	VDE mark (Note: Only appears on parts ordered with VDE option – See order entry table)
4	One or two digit year code • Two digits for black package parts, e.g., '03' • One digit for white package parts, e.g., '3'
5	Two digit work week ranging from '01' to '53'
6	Assembly package code

*Note – Parts built in the white package (M suffix) that do not have the 'V' option (see definition 3 above) that are marked with date code '325' or earlier are marked in the portrait format.

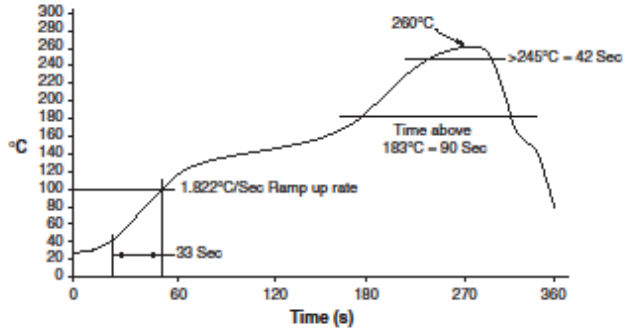
QT Carrier Tape Specifications (Black Package, No Suffix)



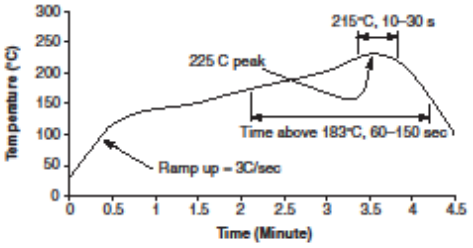
QT Carrier Tape Specifications (White Package, -M Suffix)



Reflow Profile (White Package, -M Suffix)



Reflow Profile (Black Package, No Suffix)



- Peak reflow temperature: 225°C (package surface temperature)
- Time of temperature higher than 183°C for 60–150 seconds
- One time soldering reflow is recommended

DISCLAIMER

FAIRCHILD SEMICONDUCTOR RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN TO IMPROVE RELIABILITY, FUNCTION OR DESIGN. FAIRCHILD DOES NOT ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT DESCRIBED HEREIN; NEITHER DOES IT CONVEY ANY LICENSE UNDER ITS PATENT RIGHTS, NOR THE RIGHTS OF OTHERS.

LIFE SUPPORT POLICY

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury of the user.
2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.