UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Escalonamento em Redes Ethernet Industrial

Humberto Figueiredo de Carvalho

Itajubá, Junho de 2007

2007	Humberto Figueiredo de Carvalho	Dissertação de Mestrado

UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Humberto Figueiredo de Carvalho

Escalonamento em Redes Ethernet Industrial

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica como parte dos requisitos para obtenção do Título de Mestre em Ciências em Engenharia Elétrica.

Área de Concentração: Automação e Sistemas Elétricos Industriais

Orientadora: Profa. Dra. Lúcia Regina Horta Rodrigues Franco

Junho de 2007

Itajubá - MG

Agradecimento

Agradeço primeiramente a Deus pelo dom da vida e por mais esta oportunidade concedida na realização deste trabalho.

A meus pais por toda a motivação e ajuda até aqui.

A minha namorada Fernanda da qual furtei preciosas horas durante os últimos anos para me dedicar ao desenvolvimento deste trabalho.

Agradeço a todos os amigos da empresa Sense Eletrônica Ltda. que me auxiliaram e me concederam a possibilidade da realização deste estudo, principalmente Sergio Augusto Bertoloni e Alexandre Baratela Lugli.

A minha orientadora Profa. Dra. Lúcia Regina Horta Rodrigues Franco, pela paciência e atenção em suprir minhas necessidades de conhecimento e experiência, na análise e correção do texto da dissertação, bem como nas informações e conselhos que auxiliaram na elaboração deste trabalho.

Agradeço também a equipe de pesquisa do LabTi da Unifei, Edson Beraldo Junior e Otávio de Souza Martins Gomes, pelo empenho e dedicação dispensados no desenvolvimento e nos testes dos softwares.

De forma geral, gostaria de agradecer a todos que de forma direta ou indireta me incentivaram na realização e conclusão deste trabalho.

Resumo

A tecnologia Ethernet do nível empresarial forçou a evolução de outros padrões fieldbuses no chão-de-fábrica. Com isto, vários fabricantes apresentam atualmente diferentes soluções Ethernet agora chamada de Ethernet Industrial. Isto gerou vários padrões de vários fabricantes que se tornaram incompatíveis entre si. Assim, Ethernet industrial é o nome dado ao uso do protocolo Ethernet na indústria para automação, controle de processos, controle de máquinas entre outras aplicações no chão-de-fábrica.

Com a evolução da Ethernet no chão-de-fábrica, sistemas de controle podem ter uma maior velocidade, um menor tempo de resposta porque a Ethernet é mais rápida que qualquer outro fieldbus.

Este trabalho propõe a implementação de um software escalonador de comunicação para uma rede Ethernet Industrial com baixo tempo de ciclo, como controle de movimento, por exemplo. Será analisado se o escalonamento traz benefícios na comunicação, tempo de resposta e tráfego na rede. Os resultados obtidos serão comparados com os resultados esperados levando-se em conta a velocidade da rede e tempo de processamento envolvido no sistema.

O escalonamento obtido foi testado através de simulações numa rede Ethernet com doze PCs (um mestre, dez escravos e um analisador de rede). O tempo de resposta foi analisado independentemente das interferências externas causadas pelo sistema utilizado para simulação, tais como: interferências do sistema operacional, da plataforma de desenvolvimento, do software desenvolvido, do hardware utilizado e tempos de processamento.

O método de comunicação utilizado será a comunicação UDP/IP que é mais utilizada para a troca de dados nos protocolos industriais.

Abstract

The Ethernet technology from enterprise level forced the evolution of other fieldbuses in the factory ground. Therefore, several manufacturers nowadays present different Ethernet solutions now called Industrial Ethernet. This engendered several patterns of several manufacturers that became incompatible among themselves. Then, Industrial Ethernet is the attributed name to Ethernet protocol in the industry for automation, process control, machines control and other factory ground applications.

With the evolution of Ethernet on factory ground, control system can have higher speed, less response time because the Ethernet is faster than other any fieldbus.

This work proposes the implementation of communication scheduler software to an industrial Ethernet with low response time like motion control, for example. It will be analyzed if the scheduling brings to forth any benefit in communication, response time and network's traffic. The obtained results will be compared with expected results considering the network speed and processing time involved in the system.

The scheduling obtained was tested through simulations in an Ethernet network with twelve PCs (one master, ten slaves and one network analyzer). Response time was analyzed independently of the external interferences caused by the system used for simulation, such as interferences from operational system, development platform, developed software, used hardware and processing time.

The communication method used will be UDP/IP that it is the most used to data exchange in industrial protocols.

Lista de Figuras

Lista de Tabelas

Lista de Siglas

ACK = Acknowledgement
ARP = Address Resolution Protocol
CAT5 = Categoria 5
CIP = Common Industrial Protocol
CLP = Controlador Lógico Programável
CPF = Communication Profile Family
CPU = Central Processing Unit
CRC = Cyclic redundancy check
CSMA/CD = Carrier Sense Multiple Access with Collision Detection
DC = Direct Current
DCCP = Datagram Congestion Control Protocol
DF = Don't Fragment
DP = Decentralized Peripherals
ED = Ethernet Device
EDS = Electronic Data Sheet
EMC = Electromagnetic Compatibility
EPA = Ethernet for Plant Automation
EPL = Ethernet Powerlink
ERP = Enterprise Resource Planning
FIN = Finish
FIP = French national standard
Gbps = Giga bits por segundo
GD = Gateway Device
GSD = General Station Description
H1 = Rede Fieldbus Foundation da Foundation
HD = Host Device
HSE = High Speed Ethernet
HMI = Human Machine Interface
Hz = hertz, oscilações por segundo [s^{-1}] ou [1/s].
ICMP = Internet control message protocol
ID = Identifier, identificador

IEC = International Electrotechnical Commission

I/O = Input / Output

IP = Internet Protocol

IP65 = Ingress Protection (IP): proteção contra penetração e jatos de água

IP67 = Ingress Protection (IP): proteção contra penetração e submersão a 1 metro de água

IPv4 = Versão 4 do Protocolo IP

IRT = Isochronous Real Tempo - IRT

ISA = Instrument Society of America (ISA),

ISO = International Organization for Standardization

LAN = Local Area Network

LD = Link Device

mA = miliampère

MAC = Media Access Control

Mbps = Mega bits por segundo

MES = Manufacturing Execution System

MF = More Fragment

ms = Milissegundos

NA = Normalmente Aberto

NF = Normalmente Fechado

ns = Nanossegundos

ODVA = Open DeviceNet Vendors Association

OSI = Open System Interconnection

PCD = Profinet Component Description

PD = Powered Device

PID = Proportional Integral Derivative

PLC = Programmable Logic Controller, em português CLP.

PSE = Power Sourcing Equipment

PSH = Push

Q = Matriz Taxa de Transição

RCF = Requests for comment

RST = Reset

RTPS = Real-time Publisher Subscriber

SCTP = Stream Control Transmission Protocol

SRT = Soft Real Time

SYN = Synchronism

Tbit = Tempo de bit

TCP = Transmission Control Protocol

TI = Tecnologia da Informação

ToS = Type of Service

TTL = Time to Live

UDP = User Datagram Protocol

URG = Urgent Pointer

us = Microssegundos

XML = Extensible Markup Language

WAN = Wide Area Network

Sumário

1. INTRODUÇÃO

Será mostrada a evolução dos fieldbuses, as vantagens e desvantagens de cada sistema e os problemas que surgiram com esta evolução. Ao mesmo tempo, serão mostradas algumas propostas de forma a se amenizar alguns destes problemas.

1.1. A Evolução

Os primeiros Fieldbus surgiram como algo inovador na indústria de processo e de manufatura [6]. Eles prometiam acabar com a complexidade de cabos existentes além de oferecerem outras funções como diagnósticos.

Desde o início vários fabricantes lutavam para se tornarem os pioneiros no mercado [6]. Desta forma, viu-se, e ainda se vê uma grande quantidade de padrões (de vários fabricantes) para diversas soluções no chão-de-fábrica.

Neste período, a rede Ethernet estava presente no escritório e em domicílios, mas mostrava sinais de que iria chegar também ao nível de fábrica. E assim, os primeiros produtos para Ethernet Industrial foram desenvolvidos.

No início vários problemas foram detectados no ingresso da Ethernet no chão-de-fábrica. Mas após um período de aperfeiçoamento e novas implementações a Ethernet pode ser considerada confiável para este ambiente [12].

Assim como para outros fieldbuses, o controle de tráfego é muito importante para Ethernet. Em situações em que o sistema de controle com baixo tempo de ciclo esteja ocupando o mesmo barramento que uma rede interna de escritório de uma indústria, o controle do acesso ao barramento deve ser bem estruturado.

1.2. A dissertação

No segundo capítulo serão mostradas as principais redes existentes no mercado, suas principais características e semelhanças.

Em seguida, no capítulo três, serão mostrados os possíveis problemas gerados em decorrência da estruturação inadequada da comunicação.

Posteriormente, no capítulo quatro, será apresentado o modelo proposto, suas bases teóricas e os resultados. Serão mostradas as simulações feitas com os respectivos resultados no capítulo cinco.

Por fim, tendo-se levantado todos os resultados, pode-se então apresentar de forma mais enfática as contribuições deste trabalho além de possíveis desenvolvimentos que podem se originar do mesmo.

2. OS FIELDBUSES E A ETHERNET

2.1. Histórico

A rede Ethernet é mais antiga que os Fieldbuses, porém a introdução da Ethernet na industrial foi feita devido às estes Fieldbuses.

O termo fieldbus, mencionado anteriormente, é um termo genérico que descreve as redes digitais de comunicação com a finalidade de substituir os antigos padrões 4-20mA existentes [1].

Nos anos 40, a instrumentação de processo confiava em sinais de pressão de 3–15 psi para monitorar os dispositivos de controle no chão-de-fábrica [1].

Já nos anos 60 os sinais analógicos de 4-20mA foram introduzidos na indústria para medição e monitoração de dispositivos [1].

Com o desenvolvimento de processadores nos anos 70, surgiu a idéia de se utilizar computadores para monitoração de processos e se fazer o controle de um ponto central [1]. Com o uso de computadores, várias etapas do controle puderam ser feitas de diferentes forma uma das outras de modo a se adaptar mais precisamente às necessidades de cada processo.

Nos anos 80 começou-se a desenvolver os primeiros sensores inteligentes assim como os controles digitais associados a esses sensores [1]. Tendo-se os instrumentos digitais era necessário algo que pudesse interligá-los. Aqui, já se tinha a idéia de uma rede que ligasse todos os dispositivos e disponibilizasse todos os sinais do processo num mesmo meio. A partir daí, a necessidade de uma rede (fieldbus) era clara, assim como um padrão que pudesse deixá-lo compatível com o controle de instrumentos inteligentes.

A busca pela definição de um padrão internacional levou vários grupos a se unirem. Entre eles: a Instrument Society of America (ISA) [2], a International Electrotechnical Commission (IEC) [3], o comitê de padronização do Profibus (norma alemã) [4] e o comitê de padronização do FIP (norma francesa) [5] e assim formou-se o comitê IEC/ISA SP50 Fieldbus.

O desenvolvimento deste padrão internacional demorou muitos anos. Em 2000, todas as organizações interessadas convergiram para criar o fieldbus padrão IEC, que foi denominado IEC 61158 [6] com oito padrões diferentes chamados "tipos”:

- Tipo 1 — FOUNDATION Fieldbus H1
- Tipo 2 — ControlNet

- Tipo 3 — Profibus
- Tipo 4 — P - Net
- Tipo 5 — FOUNDATION Fieldbus HSE (High Speed Ethernet)
- Tipo 6 — Interbus
- Tipo 7 — SwiftNet
- Tipo 8 — WorldFIP

Mesmo estes tipos não abrangiam todas as aplicações na indústria. Mais tarde, então, foi criada a IEC 61784 [6] [7] como uma definição dos chamados "perfis" e ao mesmo tempo foram corrigidas as especificações de IEC 61158. A Tabela 1 mostra os padrões com os perfis.

IEC 61784	IEC 61158 - PROTOCOLOS		
	MEIO FISICO	CAMADA	
CPF-1/1	TIPO 1	TIPO 1	FOUNDATION FIELDBUS (H1)
CPF-1/2	ETHERNET	TCP/UDP/IP	FOUNDATION FIELDBUS (HSE)
CPF-1/3	TIPO 1	TIPO 1	FOUNDATION FIELDBUS (H2)
CPF-2/1	TIPO 2	TIPO 2	CONTROLNET
CPF-2/2	ETHERNET	TCP/UDP/IP	ETHERNET/IP
CPF-3/1	TIPO 3	TIPO 3	PROFIBUS-DP
CPF-3/2	TIPO 1	TIPO 3	PROFIBUS-PA
CPF-3/3	ETHERNET	TCP/UDP/IP	PROFINET
CPF-4/1	TIPO 4	TIPO 4	P-NET RS-485
CPF-4/1	TIPO 4	TIPO 4	P-NET RS-232
CPF-5/1	TIPO 1	TIPO 7	WORLDFIP(MPS,MCS)
CPF-5/2	TIPO 1	TIPO 7	WORLDFIP(MPS,MCS, SubMMS)
CPF-5/3	TIPO 1	TIPO 7	WORLDFIP(MPS)
CPF-6/1	TIPO 8	TIPO 8	INTERBUS
CPF-6/2	TIPO 8	TIPO 8	INTERBUS TCP/IP
CPF-6/3	TIPO 8	TIPO 8	INTERBUS SUBSET
CPF-7/1	TIPO 6	TIPO 6	SWIFTNET TRANSPORT
CPF-7/2	TIPO 6	TIPO 6	SWIFTNET FULL STACK

Tabela 1 - Padrões e protocolos de acordo com a IEC 61784 e IEC 61158 [6]

Como pode ser observado na Tabela 1, vários protocolos de Ethernet já foram incluídos. Estes protocolos usam o meio físico da Ethernet bem como os protocolos IP, TCP e UDP.

2.2. A Ethernet na Indústria

Há vários fieldbuses no ambiente industrial. Devicenet [7] [9] [56] , Profibus [7] [8], Interbus [7], Fieldbus Foundation [7] entre outros são usados em muitas aplicações. Todos podem ser usados de acordo com a preferência e, às vezes, com a aplicação.

Atualmente, cada fabricante já tem sua solução para o ambiente industrial em Ethernet: Profinet [7] [8] da associação Profibus é uma evolução do Profibus-DP, Ethernet/IP da

ODVA [7] [9] é uma evolução do Devicenet e Controlnet e HSE High Speed Ethernet [7] da Fieldbus Foundation (que interconecta as redes H1) são exemplos e padrões, conforme apresentados na Tabela 1.

Com a existência de uma grande quantidade de soluções para Ethernet Industrial, acabou-se por não ter a interoperabilidade desejada. Isto porque cada fabricante ou grupo desenvolveu suas soluções incompatíveis com os demais, por exemplo, o Profinet da associação Profibus não se comunica com o Ethernet/IP da ODVA.

De qualquer forma a Ethernet conseguiu sua penetração no ambiente industrial, porém alguns problemas começaram a surgir nesta fase inicial.

2.3. Switch como solução para o determinismo

No principio a Ethernet não foi considerada ideal para aplicações industrial por não ser determinística [6]. Na estratégia de acesso ao meio CSMA/CD [10], as colisões são detectadas e em seguida há contagem de tempo aleatória para uma nova transmissão. Este método não parecia uma solução muito atraente para a indústria porque não se garantia que os dados fossem realmente transmitidos. Podem ocorrer várias colisões sucessivas e algumas informações podem não ser transmitidas e perder sua validade. O uso do switch amenizou este problema [6].

O switch é composto de várias portas com buffer mantendo o controle de colisão especificada no método de acesso ao meio CSMA/CD. Se houver duas transmissões simultâneas, como o switch tem portas independentes, pode-se transmitir o frame de uma porta e armazenar o frame da outra em um buffer a ser transmitido posteriormente. Assim, assegura-se que sempre um dado transmitido chegue ao seu destino e o tempo necessário para isto. Desta forma, a Ethernet estava mais apta a ser utilizada no chão-de-fábrica [11].

Mesmo com o uso de switchs, eventuais colisões poderiam acontecer: se houvesse um broadcasting no switch, poderia haver eventuais colisões com um dispositivo transmitindo numa mesma porta. Para resolver mais este problema, o sistema full duplex foi aplicado, no qual há um canal de transmissão e um canal de recepção evitando assim a situação citada anteriormente, pois mesmo que o switch transmita um broadcasting e um dispositivo neste mesmo instante transmita outra informação, eles estariam fisicamente em conexões diferentes e desta forma não ocorreria colisão. Assim, a Ethernet pode ser considerada mais aplicável ao ambiente industrial [12].

2.4. As vantagens e desvantagens da Ethernet na Indústria

A Ethernet de escritório que está sendo levado ao chão-de-fábrica traz inúmeras vantagens agregadas. Por ser um padrão já consolidado no mercado, não há maiores problemas de aceitação. A utilização dos conceitos e equipamentos já existentes no mercado (conectores, placas, cabos, etc.) torna-se um fator positivo aos envolvidos na manipulação desta nova tecnologia, que na verdade não é nova, só adaptada.

A velocidade alcançada com a Ethernet na indústria (até 1 Gbps) é algo novo visto que nenhum fieldbus existente no mercado conseguiria sequer se aproximar desta taxa.

Tudo o que for usado na indústria, pode ser usado no escritório, porém a recíproca não é verdadeira. Um dispositivo simples de escritório pode não suportar o ambiente agressivo do chão-de-fábrica.

Escritório	Indústria
Cabeamento simples	Cabeamento complexo
Cabeamento sob o chão	Cabeamento sobre o chão
Conexão de dispositivos variáveis em mesmo ambiente	Conexões fixas de dispositivos
Cabos e conexões pré-fabricadas	Conexões com os dispositivos têm que ser preparadas no campo
Topologia em árvore	Topologia em linha e em anel (redundante)
Grandes pacotes de dados, como imagens, sons, video,...	Pequenos pacotes de dados, como medidas, valores,...
Temperatura moderada (de 0° a 50°)	Temperaturas extremas (de -20° a 70°)
Nenhuma vibração	Vibração de máquinas
Pouco ruído	Muita influência de ruídos (EMC)
Sem problemas mecânicos	Perigo de danos mecânicos
Sem problemas químicos	Perigos químicos de atmosferas oleosas ou agressivas

Tabela 2 - Diferenças entre Ethernet na indústria e no escritório

Conforme descrito anteriormente, um grande atrativo da Ethernet na indústria é ser uma rede já consolidada e de grande penetração em todo mundo. A adequação do meio físico não apresenta impacto significativo para as pessoas que vão usá-la na indústria.

Na Tabela 2 as principais diferenças da Ethernet na indústria e no escritório são mostradas. Dentre as diferenças principais, algumas são bem significativas como a variação de temperatura e as interferências sofridas. Desta forma, um dispositivo industrial exigirá outros testes funcionais e também normas mais exigentes para se obter as certificações

necessárias para uso destes dispositivos no chão-de-fábrica. Com isto são necessários testes como vibração, temperatura, EMC em dispositivos Ethernet usados na indústria.

2.5. Modelos OSI e TCP/IP

Por ser a Ethernet Industrial derivada da Ethernet é necessário o conhecimento do modelamento desta rede.

Para um melhor entendimento e padronização da arquitetura de rede, foi desenvolvido o modelo de camadas que apresenta muitas vantagens: fácil aprendizagem, implementação mais fácil pelo uso de camadas individuais, manutenção e aperfeiçoamentos de cada camada.

Para se obter interconectividade entre máquinas de diferentes sistemas operacionais, a Organização Internacional de Padronização (ISO - International Organization for Standardization) aprovou nos anos 80 um modelo de referência para permitir a comunicação entre máquinas diferentes denominado OSI (Open Systems Interconnection) [13].

O modelo em camadas desenvolvido pela ISO foi um primeiro caminho para padronização internacional [14]. Este modelo, mostrado na Figura 1, é dividido em sete camadas com funções e características bem definidas.

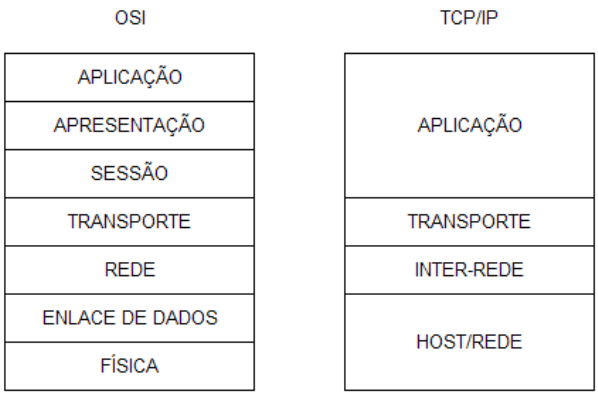

Figura 1 - Modelos OSI e TCP/IP [14]

Além do modelo OSI citado acima, há também o modelo de referência chamada modelo TCP/IP que teve origem numa rede mais antiga que é a ARPANET, um modelo de rede criado pelo Departamento de Defesa dos Estados Unidos e que foi muito utilizado em universidades [14]. Este modelo ARPANET foi definido em [15] e discutida em [16]. Nas camadas deste modelo TCP/IP podem ser feita a comparação com o modelo OSI conforme mostrado na Figura 1.

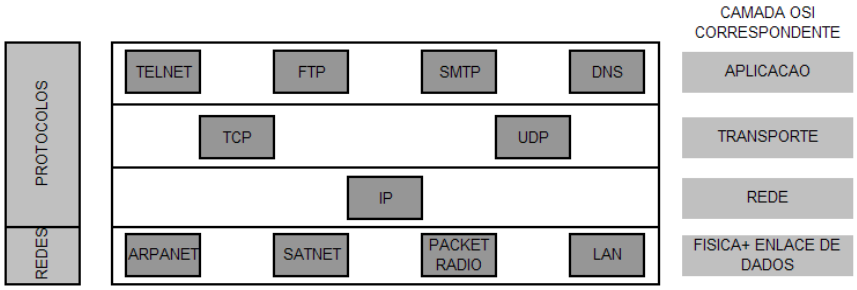

Figura 2 - Protocolos e Redes no Modelo TCP/IP [14]

Conforme a Figura 2, pode-se ter, por exemplo: no meio físico uma LAN, na camada de rede o protocolo IP, na camada de transporte o frame o TCP e uma aplicação Telnet.

Neste trabalho, estará sendo referenciado o modelo TCP/IP desde a camada mais baixa (física de host/rede) até a camada mais superior (com a aplicação).

2.6. Meio Físico

O meio físico é responsável pela transmissão de informações de uma origem ao um destino. Esta informação pode ser transmitida como bit na forma elétrica, luminosa ou eletromagnética. Na indústria estes três padrões e outros mais podem estar presentes.

A estrutura física da Ethernet é extrema importante na indústria devido ao ambiente agressivo em que se instala.

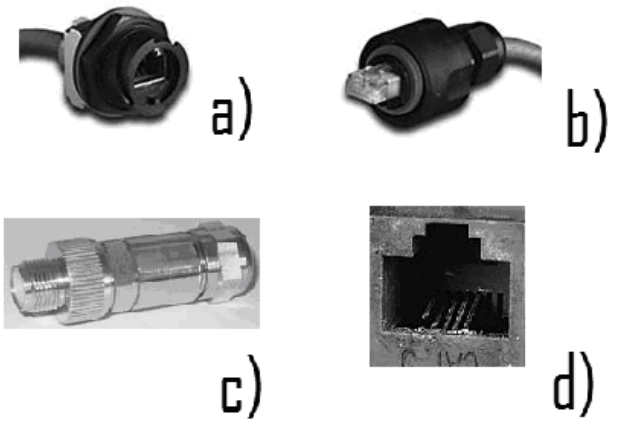

Figura 3 - Conectores [17]

Os conectores têm grande importância nesta atmosfera industrial, podendo garantir, por exemplo, a proteção de umidade, a proteção mecânica, poeira e outras situações comuns no chão-de-fábrica [17].

A Figura 3 (d) mostra um problema decorrente de corrosão.

Uma solução para estes problemas é a adoção de conectores mais robustos [18] com IP67 (à prova de imersão em água) conforme mostra a Figura 3(a), (b) e (c), com um conector M12 IP67 e um RJ 45 também IP67.

2.7. O Quadro IEEE 802.3

O IEEE 802.3 [10] é composto de vários padrões que definem o meio físico e a camada de enlace na Ethernet. Geralmente este padrão é usado em LAN ou WAN. O meio físico é usado para conectar computadores entre si ou com outras rede usando-se HUBs, switchs e roteadores, por exemplo. Esta ligação pode ser feita através de cabos de cobre ou fibra ótica.

Na camada de enlace o IEEE 802.3 estabelece o formato do quadro usado para transmissão de dados. O formato do quadro na camada de enlace é mostrado na Figura 4.

Além disto, o método de acesso ao meio CSMA/CD [10] também é descrito neste padrão.

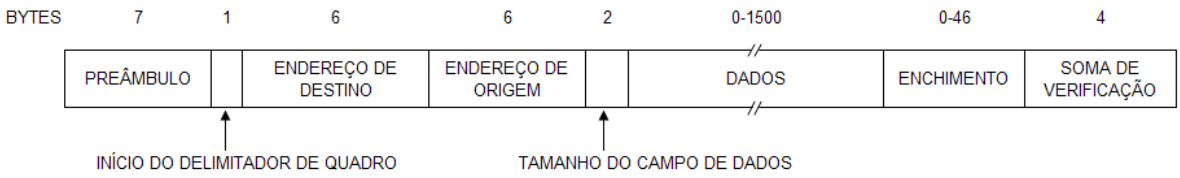

Figura 4 - Formato do quadro IEEE 802.3 [14]

Têm-se os seguintes campos:

- Preâmbulo (7 bytes = 56 bits): responsável pela sincronização do receptor com o sinal. Após o preâmbulo há um byte de início de quadro.
- Endereço destino (6 bytes = 48 bits) e o Endereço origem (6 bytes = 48 bits) são os respectivos endereços MAC de destino e de origem dos dados.
- Tamanho do campo de dados indica quantos dados estarão presentes no campo de dados. O valor é de 0 a 1500 bytes. Se os dados forem menores que 46 bytes, bytes de enchimento serão utilizados até o máximo de 46 bytes.
- Em seguida têm-se os dados a serem transmitidos. Dentro destes dados pode haver um protocolo IP, ARP, etc..
- No final, há o campo CRC (cyclic redundancy check). É um código de verificação de 32 bits que faz a verificação do cabeçalho e dos dados.

Sem o preâmbulo o tamanho máximo do quadro é de 1518 bytes e o tamanho mínimo é de 64 bytes.

O motivo para se ter um tamanho mínimo do quadro é evitar que um dispositivo termine a transmissão de um dado curto sem este ter chegado à extremidade do cabo [15] onde poderia ocorrer uma colisão. Para isto, caso seja necessário, dados de preenchimento de 0-46 bytes devem ser adicionado no campo de dados para se atingir o tamanho mínimo.

2.8. O Frame IP

Dentro do campo de dados do IEEE 802.3 mostrado na Figura 4, pode ser utilizado o protocolo IP [19], como no caso da Ethernet Industrial.

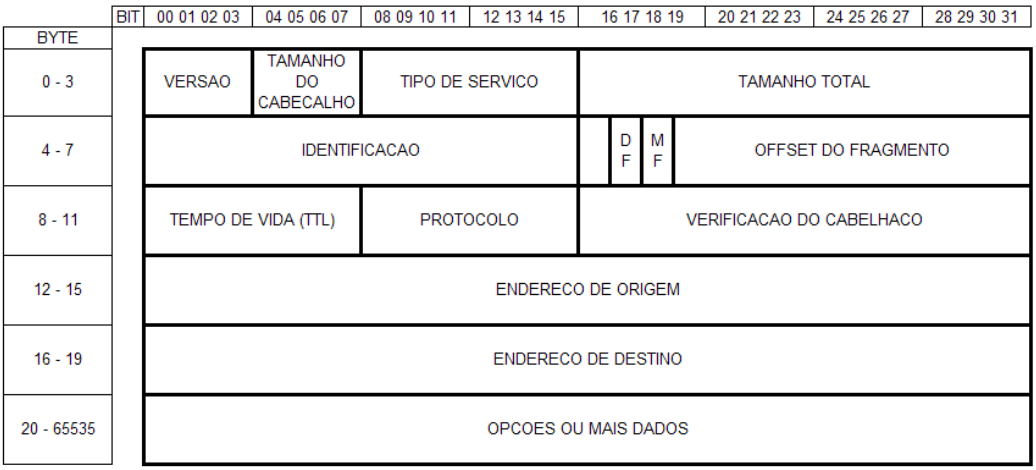

Figura 5 - Cabeçalho IP [14]

O frame IP, mostrado na Figura 5, tem os seguintes campos:

- Versão: indica a versão do protocolo.
- Tamanho do Cabeçalho.
- Tipo de Serviço.
- Tamanho Total: tamanho, em bytes, de todo o datagrama, incluindo cabeçalho e dados. O tamanho mínimo do datagrama é 20 bytes e o máximo é de 65535 bytes.
- Identificador: usado para identificar fragmentos do datagrama IP original.
- Flags DF e MF: DF (Don't Fragment) é uma indicação para que o roteador não fragmente o quadro IP. MF (More Fragment) é colocado em todos os frames fragmentados exceto o último. Isto é necessário para se determinar o termino dos pacotes fragmentados.

- Offset do fragmento: permite que um receptor determine o número de um fragmento no datagrama IP original.
- Tempo de Vida: TTL (Time To Live) limita a vida de um datagrama em contagem de saltos. Quando um pacote atravessa um router, o TTL é decrementado e quando este campo chega ao zero o pacote é descartado.
- Protocolo: Este campo define o protocolo usado no campo de dados de um datagrama IP, podendo ser ICMP, o TCP ou o UDP, por exemplo.
- Verificação do cabeçalho: é checksum para o cabeçalho do datagrama. Envolve apenas verificação do cabeçalho (os dados não são verificados).
- Endereço de origem/destino: endereço IP de origem e de endereço IP de destino dos dados no frame.
- Opções: Campo adicional do cabeçalho, mas normalmente não são usados.
- Dados: dados que podem contém inclusive outro protocolo. Tamanho máximo é de 65515 bytes.

2.9. O Protocolo TCP

No quadro IEEE 802.3, pode estar encapsulado dentro do protocolo IP [19] o protocolo TCP (Transmission Control Protocol) [20], ou o UDP (User Datagram Protocol) [21], ou o SCTP (Stream Control Transmission Protocol) [22], ou o DCCP (Datagram Congestion Control Protocol) [23] entre outros.

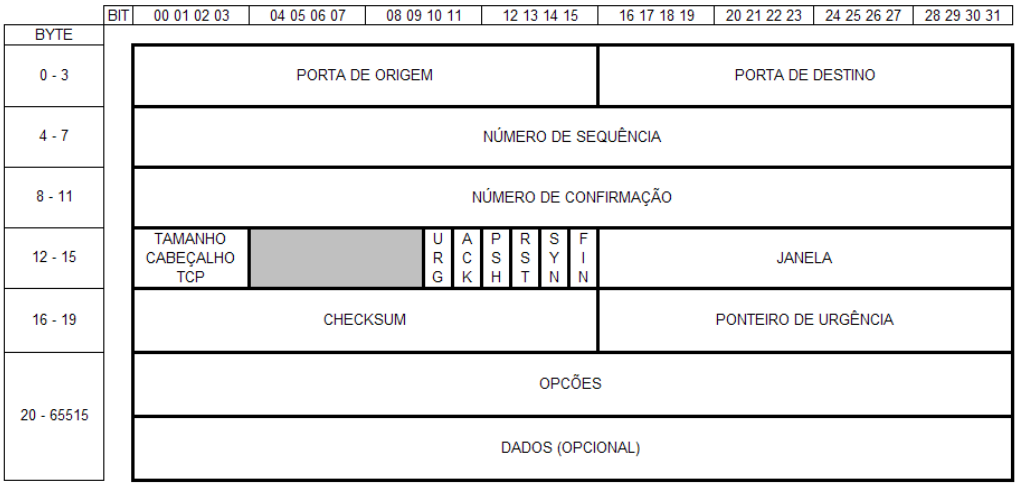

Figura 6 – Formato do Quadro TCP [14]

O Transmission Control Protocol (TCP) é um protocolo da camada de transporte orientado à conexão, que fornece transmissão confiável de dados full duplex. O TCP é parte da pilha de protocolos do modelo TCP/IP.

Conforme será detalhado posteriormente, os mais usados na Ethernet Industrial são os TCP e o UDP.

A Figura 6 mostra o formato de um quadro TCP. O TCP tem um cabeçalho fixo de 20 bytes e pode ser seguido por opções do cabelhaço. Depois das opções poderá haver até 65.495 bytes de dados. Este número é retirado do tamanho total do IP menos cabeçalho do IP e do TCP [14]:

65.535 (IP total) – 20 (cabeçalho IP) – 20 (cabeçalho TCP) = 65.495

A estrutura do TCP mostrada na Figura 6 é a seguinte:

- Os campos Porta de Origem e de Destino indicam as conexões estabelecidas.
- Número de Seqüência e Número de Confirmação, respectivamente, é o número usado para garantir a seqüência correta dos dados de chegada e o próximo byte TCP esperado.
- Tamanho do cabeçalho TCP em bytes.
- Seis flags com funções de controle (como a configuração e a terminação de uma sessão) que são URG (Urgent Pointer), ACK (Acknowledgement), PSH (Push), RST (Reset), SYN (synchronism) e FIN (Finish).
- O campo Janela é utilizado para controle de fluxo.
- Checksum faz a verificação do cabeçalho e dos campos de dados.
- O Ponteiro de urgência indica o final dos dados urgentes.
- O campo opções oferece alguns recursos extras que não foram inicialmente previstos no cabeçalho.
- O campo Dados contém os dados do protocolo da camada superior.

2.10. O Protocolo UDP

O UDP (User Datagram Protocol) é o protocolo de transporte do modelo TCP/IP não orientado à conexão. O UDP é um protocolo simples que troca datagramas, sem confirmações ou entrega garantida. O processamento de erros e a retransmissão de erros devem ser tratados por outros protocolos [14].

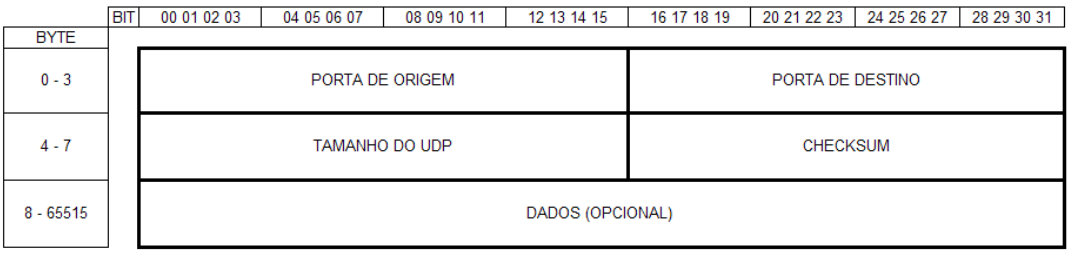

Figura 7 - Formato do Quadro UDP [14]

O UDP não usa janelamento ou confirmações, portanto cabe aos protocolos da camada de aplicação fornecer esta confiabilidade.

Os campos do UDP são:

- Porta de origem e de destino: identificam os pontos finais nas máquinas de origem e destino.
- Tamanha da Mensagem UDP: tamanha em bytes do datagrama (cabeçalho e dados). O tamanho mínimo é de 8 bytes, que é o tamanho do cabeçalho, e o tamanho máximo é teoricamente de 65515 bytes, com 65507 de dados.
- Checksum UDP: checksum calculado do cabeçalho e dos campos de dados

2.11. Protocolos utilizados na Ethernet industrial

Em uma indústria de controle que use Ethernet industrial, os dados devem ser lidos dentro de um intervalo de tempo aceitável para a aplicação.

O protocolo UDP é mais rápido que o protocolo TCP, pois o UDP não é orientado a conexão, não verificando a confirmação de recepção e também exigindo menos tempo de processamento por ter um cabeçalho menor. Por isto, para troca de dados de dispositivos de entrada e saída o UDP é o protocolo mais utilizado.

Eventualmente a não confirmação de recepção dos dados poderia acarretar muitas informações recebidas erroneamente. Isto realmente pode acontecer. Porém, se um dado for verificado como incorreto, seria necessário uma retransmissão. Numa rede Ethernet a velocidade da atualização dos dados é muito alta e um dado transmitido erroneamente já estaria obsoleto para uma retransmissão. Desta forma, as retransmissões não apresentariam nenhuma vantagem e por isto o UDP é usado e se evidencia que a retransmissão não é necessária.

O protocolo TCP é orientado à conexão e tem tratamentos para retransmissão no caso de perca de dados. Na Ethernet industrial ele é usado para configuração de parâmetros. Isto é

necessário porque, diferentemente do UDP, quando um dado é transmitido ele deve chegar ao receptor. Se houver uma falha estes dados devem ser retransmitido, pois parâmetros não se tornam obsoletos e deve-se ter a garantia de recepção. Um exemplo deste tipo de parâmetro seria, por exemplo, a configuração de uma saída em NA (Normalmente Aberta) ou NF (Normalmente Fechada). Nota-se claramente que se este dado não chegar ao seu destino, um dispositivo pode ter em sua saída um estado contrário às configurações pré-estabelecidas. Além disto, o TCP, por ser maior e exigir mais processamento, não deve ser usado constantemente para que não se comprometa a comunicação de dados na rede. É exatamente o caso da configuração dos parâmetros onde a configuração é feita na instalação da rede, após isto ela é raramente executada.

2.12. A Ethernet no mercado atual

Na seqüência serão mostrados os principais protocolos existentes no mercado e a interação que foi realizada com IP, TCP e UDP.

2.12.1. Ethernet/IP

Ethernet Industrial Protocol (Ethernet/IP) [24] é um padrão de rede industrial aberto que suporta mensagem em tempo real e troca de mensagens. O Ethernet/IP usa o chip de comunicação Ethernet padrão e também o mesmo meio físico.

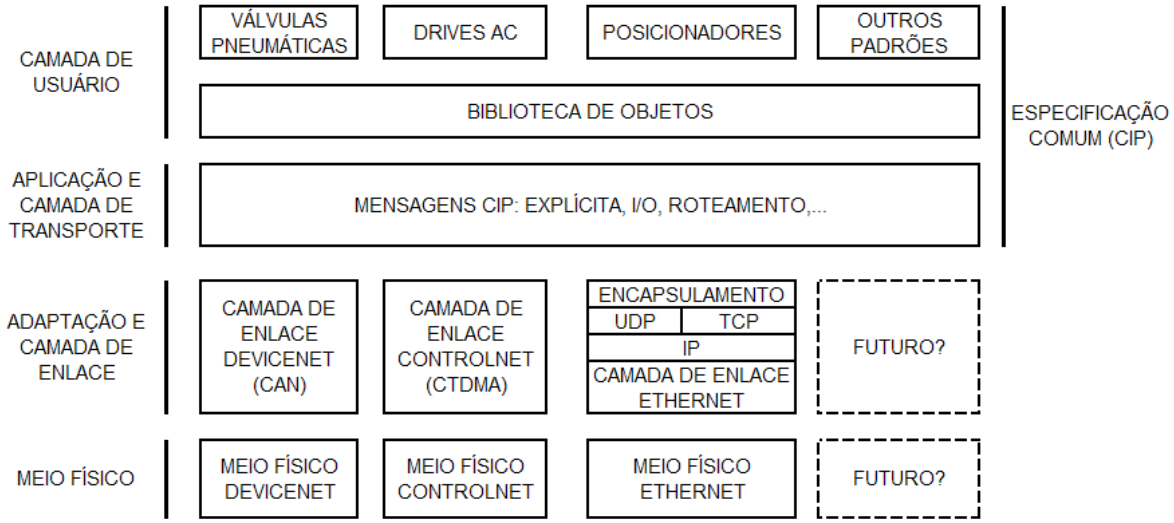

Figura 8 – CIP no Devicenet, ControlNet e Ethernet/IP [24]

Ethernet/IP é uma rede aberta baseada em:

- IEEE 802.3 padrões Físicos e enlace de dados.
- Ethernet TCP/IP (Transmission Control Protocol/Internet Protocol).
- Common Industrial Protocol – CIP, mostrada na Figura 8, é o protocolo de aplicação presente nas redes ControlNet, Devicenet e Ethernet/IP [24].

2.12.1.1. Comunicação

A CIP provê uma grande quantidade de padrões e serviços de acesso de dados e para controle de dispositivos na rede via mensagens "implícitas" (controle em tempo real) e "explícitas" mostradas na Figura 9. O pacote de dados CIP pode ser encapsulado antes que eles sejam enviados via Ethernet e é inserido um cabeçalho no datagrama que dependerá das características do serviço.

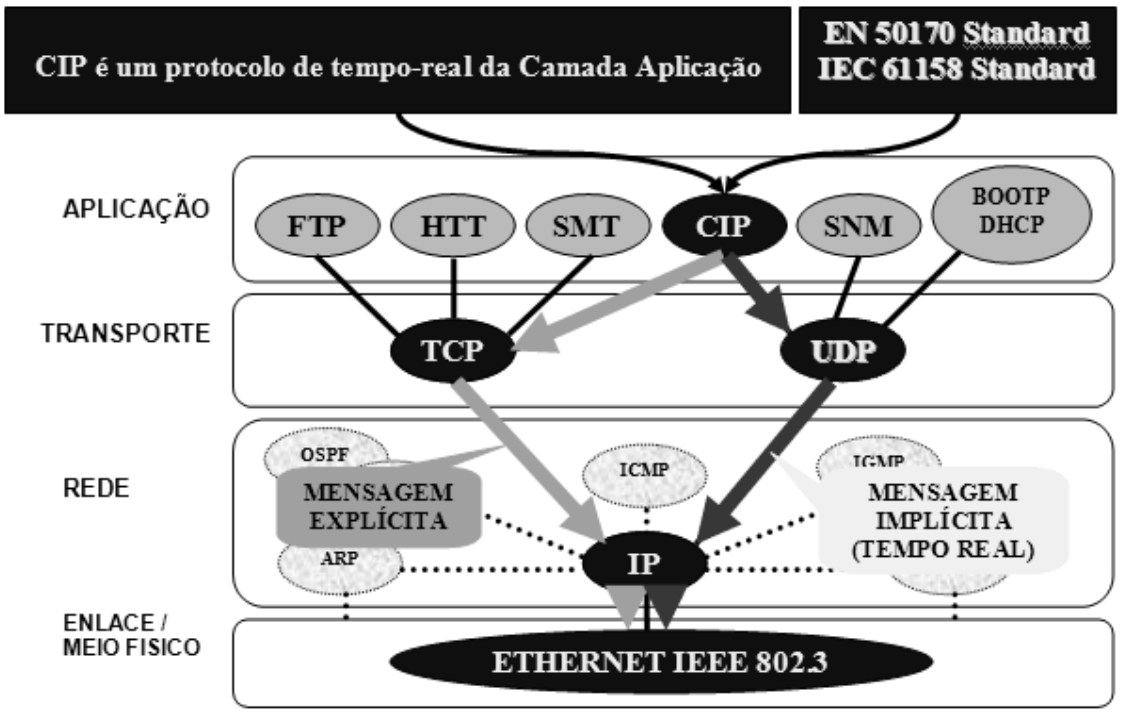

Figura 9 - Comunicação no Ethernet/IP [24]

2.12.1.2. A CIP encapsulada no UDP e TCP

A Figura 9 mostra como os dados são enviados para rede usando-se o protocolo UDP ou o protocolo TCP.

- A Transferência de dados não crítica – tipicamente pacotes grandes, conexões explícitas de um produtor para um consumidor. Os pacotes de Informações usam o

protocolo TCP/IP e têm a vantagem do tratamento de dados do TCP, ou seja, sendo orientados à conexão garantem o envio e o recebimento dos dados.

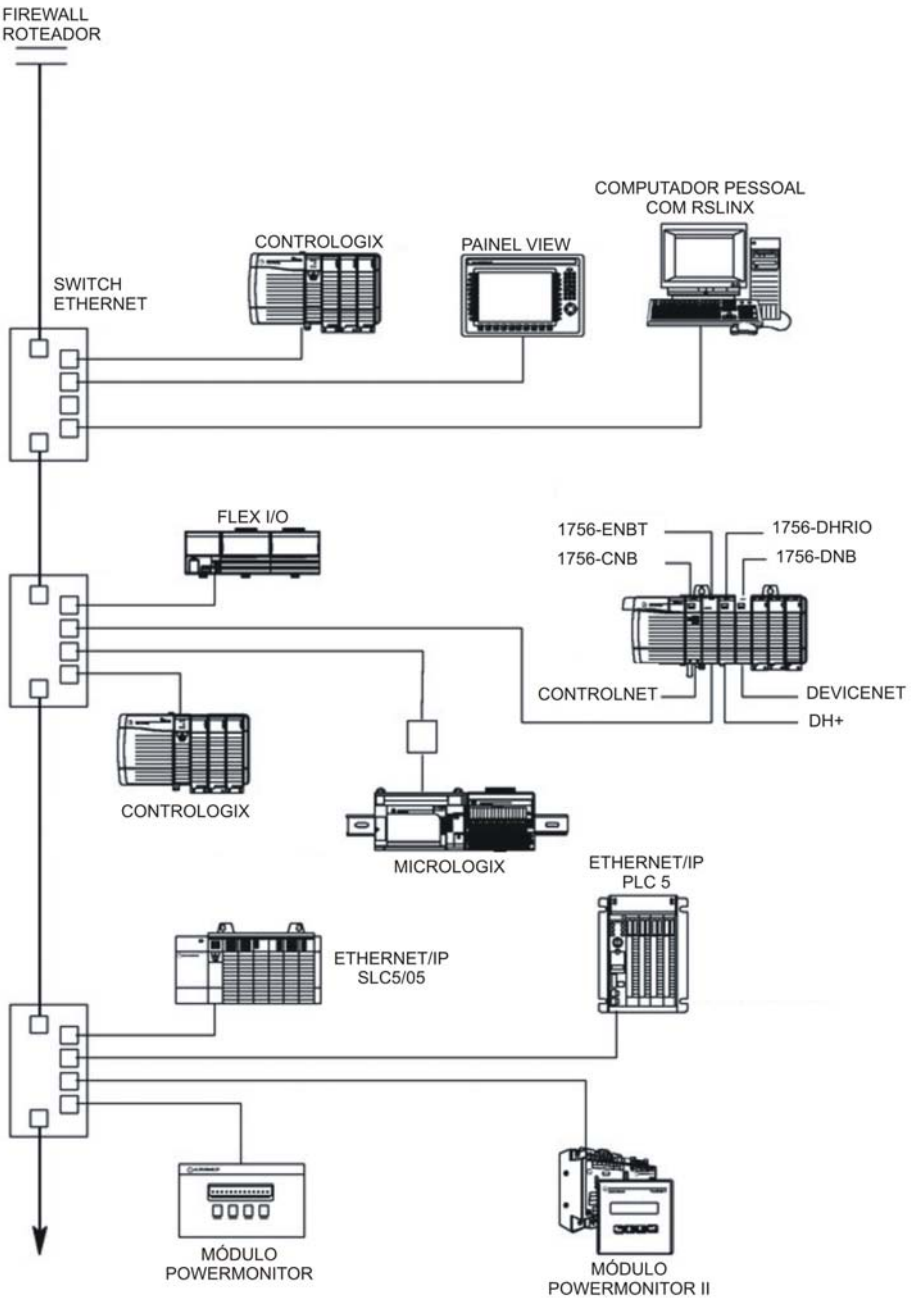

Figura 10 - Configuração Típica do Ethernet/IP [25]

- Os Dados de I/O (Input/Output) usam transferência crítica de dados, tipicamente pacotes de dados pequenos. Troca de dados I/O são conexões implícitas de longo alcance entre um produtor e um consumidor. Pacotes de Dados de I/O usam o protocolo UDP/IP e têm a vantagem da alta velocidade do UDP. Neste caso não há verificação de recepção, uma vez que, conforme discutido no início deste trabalho, qualquer dados que seja retransmitido na Ethernet já estaria obsoleto.

- Há também a sincronização em Tempo-Real que é uma **s**incronização cíclica de dados entre um produtor e um consumidor ou consumidores. Os pacotes de Sincronização em Tempo-Real usam o protocolo UDP/IP. Como são dados de sincronismo, a velocidade oferecida pelo UDP é necessária.

Na aplicação da Figura 10, todos os dispositivos têm conexão com a rede Ethernet/IP [25].

Na utilização de Flex I/Os, por exemplo, esta família de dispositivos [25] tem interfaces que podem se comunicar com Devicenet (ODVA) e Ethernet/IP. Assim, as antigas interfaces Devicenet podem ser trocadas por interfaces Ethernet/IP num mesmo cartão Flex I/O.

2.12.2. Profinet

Profinet é um padrão de automação da associação PROFIBUS internacional para implementação e integração de soluções baseadas em Ethernet Industrial [4]. O Profinet suporta a integração de um simples dispositivo de campo e aplicações de tempo crítico à comunicações Ethernet, bem como a integração de automação de sistemas distribuídos baseados em componentes [26].

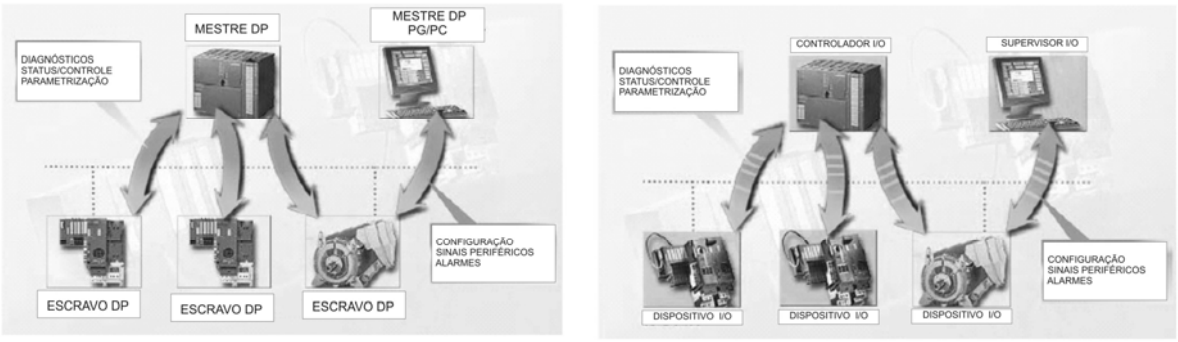

Figura 11 - Profibus x Profinet [27]

Profinet I/O distingue-se em três tipos de dispositivo: Controlador I/O, Dispositivo I/O e Supervisor I/O:

- Controlador I/O: Controlador no qual o programa de controle é executado.
- Dispositivo I/O: Dispositivo de campo remoto que é designado para um Controlador I/O.
- Supervisor IO: Dispositivo PC programável que comissiona e tem funções de diagnósticos

A arquitetura do Profinet é similar ao do Profibus DP, incluindo a comunicação mestre-escravo. O mestre DP corresponde ao controlador I/O no Profinet [27].

2.12.2.1. Dispositivos distribuídos de campo (Profinet I/O)

Dispositivos distribuídos de campo são integrados através do Profinet I/O. Ele usa a visão convencional do I/O do Profibus DP, de acordo com o qual os dados do I/O do dispositivo de campo são ciclicamente transmitidos para a imagem do CLP.

Profinet I/O [26] descreve um modelo de dispositivo que é baseado em características chaves do Profibus DP e inclui slots e canais. As características dos dispositivos de campo são descritas via um GSD (General Station Description) [28] em uma base XML.

2.12.2.2. Interação com outros Fieldbuses

Para a integração de outros dispositivos e outros fieldbus, o Profinet tem o Proxy que funciona como um gateway e desta forma transfere dados deste fieldbuses para o Profinet onde o controle é implementado. Existe um padrão especifico para o Proxy podendo ser, por exemplo, para Profibus DP ou Proxy para interbus [29] conforme mostrado na Figura 12.

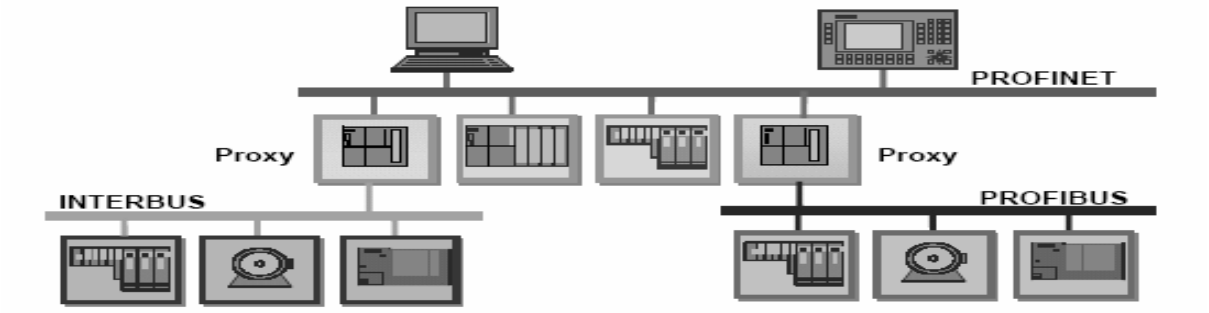

Figura 12 - Integração de Outros Fieldbuses via Proxy [27]

2.12.2.3. Comunicação

O Profinet utiliza diversos níveis de comunicação:

- Transferência dados não críticos, como parâmetros, dados de configuração e informações de conexão usando o canal padrão TCP/UDP e IP. Isto satisfaz as exigências de automação de outras redes (MES, ERP).
- Para a transmissão de dados de processo em tempo críticos usa-se o canal de tempo real - Soft Real Time (SRT).

- Para aplicações de tempo onde é necessário sincronização, a comunicação em Tempo Real Isócrona (Isochronous Real Time - IRT) está disponível permitindo jitter acumulados de 1 µs em um ciclo de 1 ms.

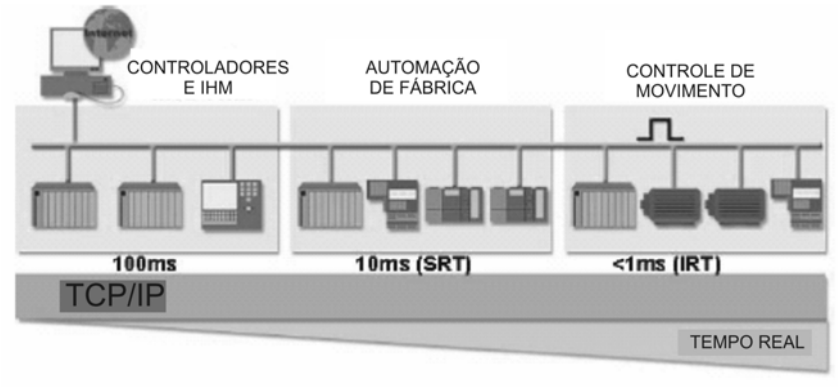

Figura 13 - Tempo de resposta [27]

De acordo com Figura 13, cada etapa do processo tem um tipo de comunicação diferente tempo exigências diferenciadas em relação ao tempo.

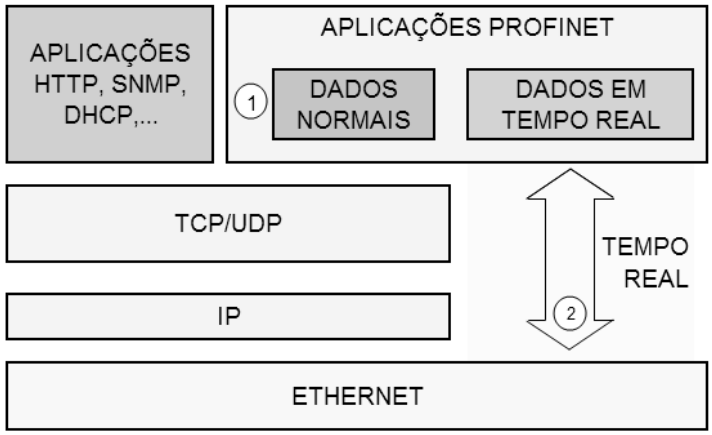

Figura 14 - Comunicação em Tempo Real [27]

Para a comunicação de dados e parâmetros que não exijam tempo de resposta crítico, o Profinet usa o TCP (Figura 13). Para dados com tempo de resposta de 10 ms (Figura 14) é usado o UDP por ser mais rápido que o TCP.

Para dados com baixo tempo de resposta, é utilizada uma comunicação especial que é chamada isócrona. Esta comunicação é feita através de um bypass na camada IP, conforme Figura 14. Assim o dado será mais rapidamente processado e enviado, pois há duas camadas a menos.

A desvantagem deste tipo de implementação é a necessidade de um hardware dedicado porque os dados em tempo real são transmitidos diretamente na Ethernet não utilizando a camada TCP/UDP nem a camada IP, conforme mostrado na Figura 14. Assim, o Profinet não é compatível com os sistemas normais Ethernet [30].

2.12.3. HSE - High Speed Ethernet

A Fieldbus Foundation tem a solução HSE para a rede Ethernet. Conforme já mostrado na Tabela 1, o H1 e o HSE são subclasses da IEC 61158.

A função do HSE não é substituir as redes H1 existentes (rede convencional Fieldbus Foundation), mas interconectá-las e ligá-las a sistemas de supervisão [31]. Esta rede usa UDP/IP sobre as camadas de enlace Ethernet [32].

	H1	HSE
Velocidade	31.25 kbps	100 Mbps
Distância	1900 m	100 m
Dois fios	Sim	Não
Multidrop	Sim	Não
Alimentação pelo barramento	Sim	Não
Segurança Intrínseca	Sim	Não
Redundância do meio	Não	Sim
Determinístico	Sim	Sim

Figura 15 - H1 e HSE [31]

Na Figura 15, podem ser observadas as diferenças entre a rede H1 (Fieldbus Foundation) e a HSE. A questão da velocidade é limitada pelo próprio limite da rede Ethernet assim como a limitação física de 100 metros do cabo da Ethernet.

O determinismo só pode ser alcançado, de acordo com o que já foi citado no início deste capítulo, pela presença do switch full duplex. Caso esta consideração não seja feita, a rede HSE, como as demais redes, não conseguiriam alcançar este "determinismo".

2.12.3.1. Categoria de dispositivos

O HSE (High Speed Ethernet) é baseado na Ethernet, IP e TCP/UDP e tem quatro tipos básicos de categorias de dispositivos (Figura 16):

- O Host Device (HD) é computador qualquer.
- O Link Device (LD) é um nó HSE para conectar um ou mais segmentos Fieldbus H1 à HSE.
- O Gateway Device (GD) é um nó HSE para conectar uma ou mais redes externas à HSE.
- O Ethernet Device (ED) é um nó HSE com condições de conexão direta às aplicações de controle e medição.

Dispositivos em Redes H1 diferentes podem se comunicar através do HSE via um Link Device (LD) (Figura 17).

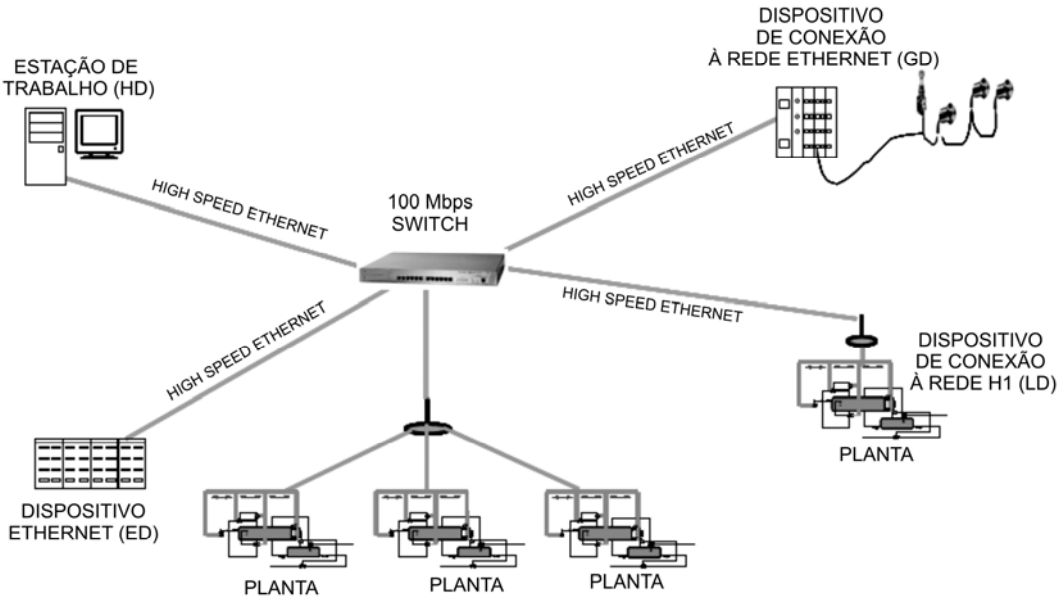

Figura 16 - Categoria de dispositivos [31]

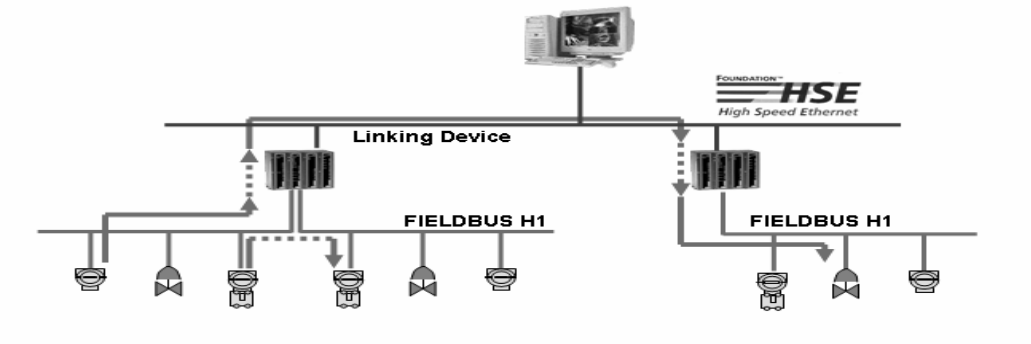

Figura 17 - Comunicação entre H1's pela HSE [31]

2.12.3.2. Comunicação

O UDP se ajusta muito bem dentro da estrutura sincronizada de cliente/servidor usado no HSE. A entrega garantida de dados é controlada pela camada de aplicação em vez da camada de transporte. Se a camada de aplicação não está adquirindo os dados que quer, tentará novamente, e se falha entra em uma ação segura.

O UDP é multicast assim pode ser usado por vários receptores em uma única comunicação. Este é tipicamente o caso para automação onde um sensor de leitura é freqüentemente usado em mais de um lugar.

2.12.4. Outros padrões na Ethernet Industrial

Além dos padrões descritos anteriormente, há vários outros, dentre os quais se pode citar:

- EPA, Ethernet for Plant Automation: é uma rede de que trabalha com os protocolos TCP/IP, seu tempo de ciclo é na ordem de 10 a 100 ms sem sincronização [33].
- Ethernet Powerlink , EPL: mostrado na Figura 18 [33], foi desenvolvido 2001 e tem como organização a EPSG (EPL Standardization Grupo).

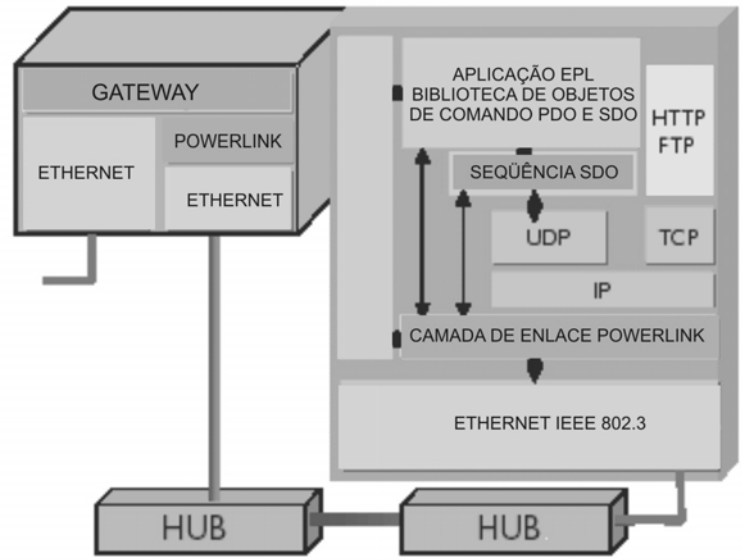

Figura 18 - Arquitetura EPL [33]

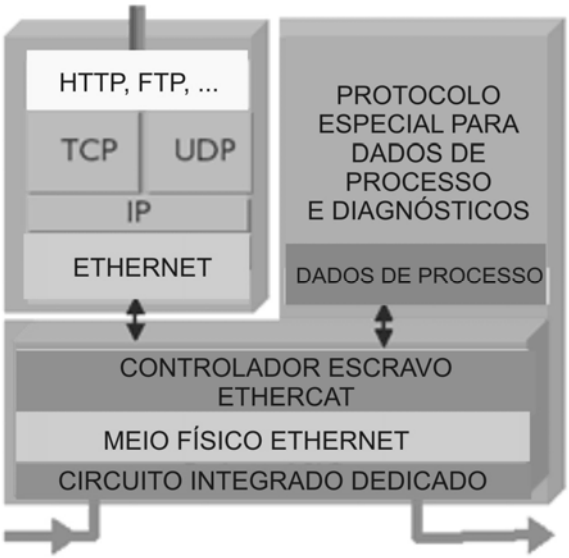

Figura 19 - Ethercat [33]

- Ethercat: mostrada na Figura 19, foi criado em 2003 [33]. A comunicação é feita através de um anel no qual o mestre envia uma mensagem que passará por todos os escravos retornando novamente ao mestre.

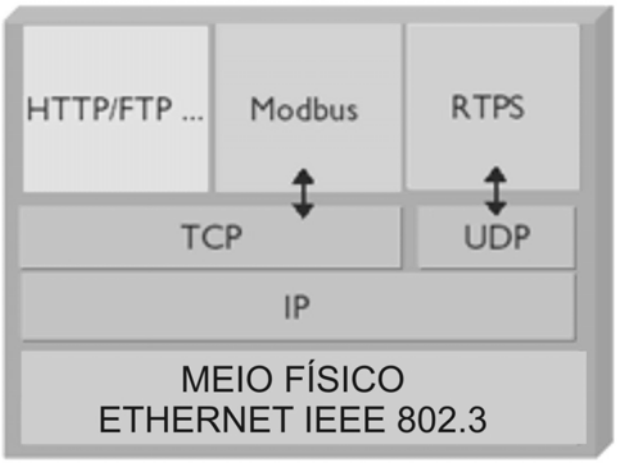

Figura 20 – Modbus/TCP [33]

- Modbus/TCP: suporta o Modbus IDA, que existe desde 1979 [33]. A comunicação em tempo real é feita usando-se o RTPS (Real-time Publisher Subscriber) encapsulado no UDP, mostrado na Figura 20, Para encapsular o protocolo Modbus existente em aplicações atuais é utilizado o encapsulado no protocolo TCP/IP.

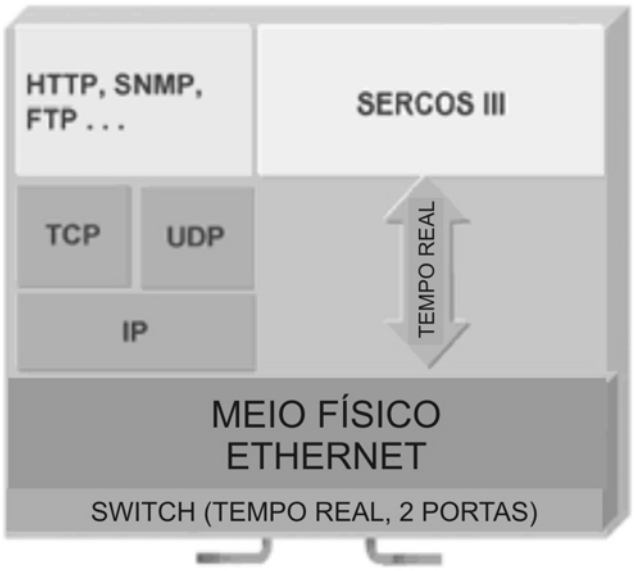

Figura 21 - Sercos III [33]

- Sercos III: é utilizado em aplicações de controle de movimento. A comunicação é em anel e há retransmissão do sinal em cada escravo da rede. O tempo de ciclo

31,25 us (com 8 nós) e 1 ms (150 nós) [33]. Um canal adicional IP (Figura 21 e Figura 22) pode ser adicionado onde é possível transferência de dados em tempo real e não-real usando os frames padrões Ethernet. Os canais cíclicos e não cíclicos podem ser configurados.

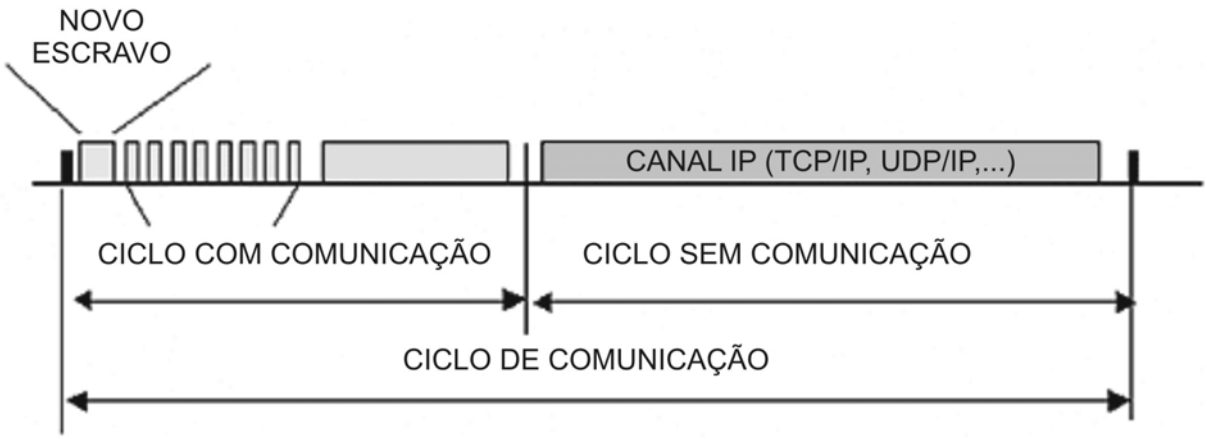

Figura 22 - Comunicação Sercos III [33]

Como pode ser observado nestes padrões acima, as soluções propostas usam também os protocolos UDP/IP e TCP/IP além de implementações diretas no frame IEEE 802.3.

3. O PROBLEMA

Nas redes Ethernet Industrial a comunicação é na maioria das vezes cíclica com atualização das tabelas de entrada e saída de um elemento de controle. Geralmente este elemento é centralizado num dispositivo de varredura da rede conectado a um CLP que terá o programa de controle da aplicação em questão.

Nos fieldbuses, antes da Ethernet, a velocidade era de algumas centenas de Kbps a alguns Mbps. A varredura da rede poderia ser de até alguns milissegundos sendo próximo do tempo de processamento da CPU do CLP.

Com a Ethernet industrial este cenário torna-se diferente. A taxa de atualização da tabela de entrada e saída do CLP é muito maior que a velocidade de processamento do CLP, em geral.

Atualmente há aplicações com tempo de ciclo pequenas o que exige também um processamento mais rápido das lógicas de controle. Assim, quando se tem uma CPU de CLP rápida a Ethernet se mostra ideal.

Em sistemas Ethernet no qual o tempo de resposta da CPU é baixo, se não houver uma estratégia de controle de tráfego pode-se ter um tráfego ineficiente na rede. Neste aspecto, o cenário para a Ethernet é diferente dos fieldbuses existentes no mercado.

Nos fieldbuses não se consegue um tempo de ciclo baixo mesmo com uma CPU rápida, pois as redes convencionais fieldbus não são tão rápidas quanto a Ethernet.

Outro fator importante é o uso de switchs para se garantir o determinismo. Cada um destes switchs terá um buffer para evitar colisões. Os buffers destes switchs são finitos e pode haver situações onde estes buffers fiquem cheios devido a tráfego excessivo e assim dados que estejam armazenados poderão ser descartados.

Trabalhos de priorização de pacotes ou portas de switchs podem ser utilizados com IEEE 802.1p [34]. Neste padrão dados com maior prioridade recebem uma identificação diferente dos demais de forma que ao chegar a um switch estes dados possam ser transmitidos antes dos demais menos prioritários.

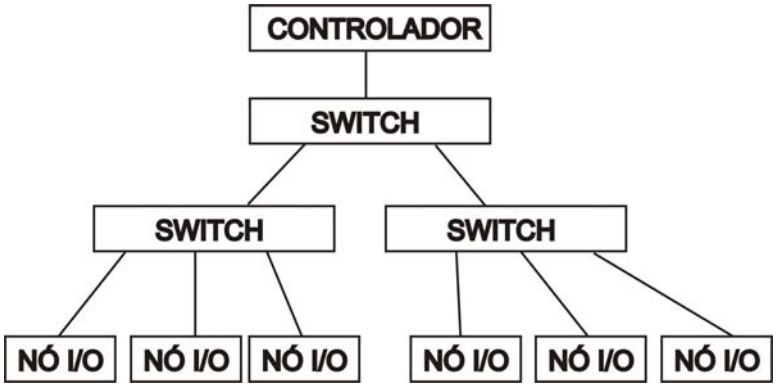

Figura 23 - Arquitetura Ethernet [35]

Quando há dados a serem enviados a vários dispositivos, podem-se usar mensagens de broadcasting ao invés da comunicação com cada dispositivo por vez [35]. Esta ação contribui para uma melhora no tráfego.

A utilização de vários switchs, conforme a Figura 23, otimiza o tráfego e exige menos a utilização de buffer de cada switch.

Usando este mesmo raciocínio, chega-se à conclusão que o projeto físico de cabeamento pode provocar um tráfego não otimizado. Isto porque, por exemplo, na Figura 23 há uma estrutura física que poderia apresentar dificuldades na instalação dos cabos. Com isto o tráfego de uma rede pode passar desnecessariamente por outra rede. Há também trabalhos [36] que mostram algumas soluções para este tipo de problema.

Independente da solução proposta há sempre um mesmo problema: o tráfego de dados na rede. Mesmo tendo-se a melhor configuração possível no barramento e o processo mais otimizado, se o tráfego for muito alto de forma a comprometer a comunicação e o tempo de ciclo, a eficiência de toda a rede é comprometida.

3.1. Trabalhos já realizados

A proposta deste trabalho, como de outros [37] [38], é o escalonamento da comunicação no barramento de forma a se utilizar somente o tempo necessário à comunicação não alterando ou até mesmo melhorando a eficiência e o tempo de ciclo do processo.

Em [39], simulações sobre tempo de transmissão e recepção foram feitas de forma a se determinar a eficiência da rede. Problemas com atrasos de transmissão gerando o atraso no controle foram feitos em [40].

Para sistemas com controle distribuído foram feitos trabalhos como [41] de forma a escalonar blocos funcionais.

Trabalhos de análise de ocupação e tempo de respostas foram feitas em [42] no qual a função do switch é comprovada conseguindo-se enviar e receber pacotes de dados mesmo ocorrendo conflitos.

Com a evolução da Ethernet, o switch é cada vez mais importante e trabalhos estão sendo feitos [43] [44] para melhorar o seu desempenho.

No trabalho de [45] foi proposto uma rede com uma maior eficiência. Esta rede usa somente o Frame Ethernet da mesma forma que o Profinet, por exemplo, não usando o IP nem o UDP de modo a se ter um menor processamento.

Uma solução para melhorar a comunicação da Ethernet no chão-de-fábrica foi proposta em [35]. As mensagens em broadcasting bem como a sincronização da comunicação são algumas opções. Na indústria como a quantidade de dados sendo menor, o uso desnecessário de mensagens pode causar estouros nos buffer de switchs.

Em [37] é utilizado o escalonamento de mensagens para que o CLP obtenha o melhor tempo de processamento através do uso de algoritmos com CPU operando como sistema multitarefa. O uso de CPU com processamento de tarefa única não foi realizado devidos aos benefícios evidentes do processo multitarefa.

Em [38] foi feito a implementação de um processamento sem multitarefa para um controle PID com I/Os remotos. Esta idéia seria a mais interessante do ponto de vista dos fielbuses que apresentam processamento centralizado. O controle no processador é executado de forma a ser o menor tempo na execução das lógicas envolvida de acordo com a temporização dos dados de entrada recebidos e dos dados de saída enviados à rede.

Nestes trabalhos citados acima, o tempo de varredura da rede é quantizado numa mesma escala de tempo do processamento da CPU do CLP. Para a rede Ethernet a relação de tempo entre a varredura da rede e o tempo de execução da CPU não deve ser quantizado como próximos.

3.2. Eficiência do Canal

Nos fieldbuses atuais, a quantidade de bytes de overhead, ou seja, os dados de cabeçalho usados para se fazer a transmissão, é pequena. Há muitos dados válidos e pouco cabelhaço para que estes dados cheguem a seu destino.

A eficiência do canal na Ethernet é mais alta para uma quantidade maior de dados conforme diz análise em [14]. De acordo com a Figura 24, quanto maior a quantidade de dados no quadro, maior a eficiência do canal.

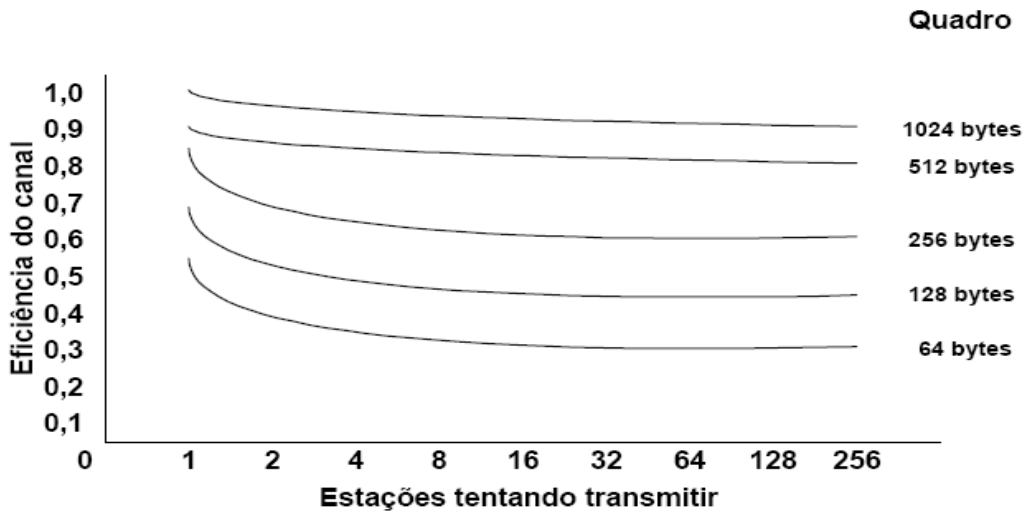

Figura 24 - Eficiência do canal do IEEE 802.3 [14]

O quadro Ethernet tem muitos bytes de cabeçalho a serem transmitidos independentes da quantidade de dados que se queira realmente transmitir.

A eficiência do canal [14] pode ser dada por:

$$\text{Eficiência do Canal: } \frac{1}{1+2BLe/cF}$$

Onde:

B = Largura de Banda

L = comprimento do cabo

e = número de slots

c = Velocidade de Propagação

F = Comprimento do quadro

Assim, quanto menor o F menor será a eficiência do canal de dados.

Se os dispositivos em uma rede tiverem uma quantidade pequena de dados de entrada e saída, o frame a ser transmitido será pequeno. Porém, o frame Ethernet terá um tamanho mínimo para trafegar no barramento.

De acordo com esta fórmula da Figura 4, a quantidade mínima do frame Ethernet será:

- 8 Bytes de preâmbulo;
- 6 Bytes de Endereço MAC de Destino;
- 6 Bytes de Endereço MAC de Origem;
- 2 Bytes do tamanho do campo de dados;
- 46 Bytes (Campo de dados onde o IP será encapsulado);

- 4 Bytes de CRC.

Total de 72 Bytes enviados.

Conforme [14] o tamanho mínimo deve ser de 64 bytes (desconsiderando-se o preâmbulo). Dados de enchimento são usados de 0 até 46 bytes de forma a se alcançar o valor mínimo de 46 bytes no campo de dados.

Apenas um byte de dado útil pode ser enviado, pois ele será encapsulado no protocolo de aplicação usando o UDP que por sua vez estará encapsulado no IP e finalmente será encapsulado no campo de dados do frame IEEE 802.3 da Figura 4.

Pode-se tomar como exemplo o uso do protocolo UDP para transferência de quatro bytes:

Utilizando o UDP (com quatro bytes), da Figura 7:

- 8 Bytes (cabeçalho= 64 bits) + 4 Bytes (4 bytes que serão transmitidos);
- Total de 12 Bytes.

Estes bytes estarão no campo de dados do IP da Figura 5:

- 20 Bytes (Cabeçalho) + 12 Bytes (protocolo UDP+dados);
- Total de 32 bytes.

Como o valor encontrado é menor que 46 bytes, o frame deverá ter bytes de preenchimento de forma a se ter 46 bytes.

O frame Ethernet IEEE 802.3 da Figura 4 terá então:

- 26 Bytes (cabeçalho + preâmbulo) + 32 Bytes (Dados = IP encapsulado);
- 14 Bytes (Dados de enchimento);
- Total de 72 Bytes enviados;

Os dados de enchimento é três vezes maior que os dados transmitidos.

Comparando-se a taxa transmitida pela taxa real de dados, têm-se:

Taxa de 100 Mbps = > tempo de bit de 10 nanossegundos

- Tempo para se transmitir um byte = 80 nanossegundos;
- Tempo para se transmitir 72 bytes (4 bytes) = 5,76 microssegundos.

Ou seja, para se transmitir 4 bytes (32 bits) o tempo necessário foi de 5,76 microssegundos. Calculando-se a taxa real de dados tem-se:

- Taxa real = 1/ (5,76u/32) = 5,55 Mbps (ou 5,55% de 100 Mbps).

Usando os dados diretamente no UDP, tem-se:

46 bytes – 20 bytes (cabeçalho IP) – 8 bytes (cabeçalho UDP) = 18 bytes

A máxima eficiência do canal se dará com o frame IP com 1500 bytes, para isto seria necessário:

Total: 1526 bytes:

- 8 Bytes de preâmbulo;
- 6 Bytes de Endereço MAC de Destino;
- 6 Bytes de Endereço MAC de Origem;
- 2 Bytes do tipo;
- 1500 Bytes (protocolo IP encapsulado);
- 4 Bytes de CRC.

Destes 1500 bytes do IP tem-se:

- 1500 bytes - 20 bytes (cabeçalho IP) = 1480 bytes.

Utilizando o UDP com 1480 bytes:

- 1480 bytes - 8 Bytes (cabeçalho) = 1472 bytes.

Estes cálculos podem ser refeitos para outros protocolos que não o UDP. Porém, se for utilizado o TCP, por exemplo, a eficiência será menor visto que o cabeçalho do TCP (Figura 6) é maior que o UDP (Figura 7).

Alguns protocolos como o Profinet (Figura 14) e Sercos III (Figura 21) usam uma técnica diferente para se ter melhor eficiência do canal. Eles podem ser usados para aplicações de controle de movimento no qual é necessário uma melhor eficiência. Para isto, não se utiliza nem os protocolos UDP ou TCP e nem o IP. Há um "bypass" nestes protocolos e os dados são inseridos diretamente no frame Ethernet.

Para, por exemplo, 46 Bytes de dados (que é a quantidade mínima), estes dados serão inseridos diretamente no frame Ethernet da Figura 4:

- 8 Bytes de preâmbulo;
- 6 Bytes de Endereço MAC de Destino;
- 6 Bytes de Endereço MAC de Origem;
- 2 Bytes do tipo;
- 46 Bytes (dados);
- 4 Bytes de CRC;

E para uma eventual transferência de 1500 bytes, a eficiência do canal seria maior conforme mostra a Figura 24.

Estes cálculos anteriores não levam em conta a aplicação usada para o encapsulamento direto no Frame IEEE 802.3, logo estes valores encontrados seriam menores dependendo do protocolo de aplicação usado.

Há varias vantagem de se usar os dados diretamente no frame IEEE 802.3: maior velocidade na transmissão, maior eficiência, menor tempo de processamento, etc. Porém, uma informação neste frame não poderá chegar a camadas superiores como a camada de inter-rede (Figura 1) onde trabalham os roteadores, por exemplo. Desta forma, para uma informação deste tipo sair de uma rede interna para uma externa será necessário o encapsulamento utilizando outro aplicativo.

Por tudo o que foi mostrado e calculado, pode-se notar que apesar da rede Ethernet na indústria ser muito rápida, esta pode não apresentar uma eficiência muita grande visto que os tráfegos de dados de processos de controle raramente chegariam aos 1472 bytes calculados anteriormente. Com isto, prova-se que o controle do tráfego de dados é importante visto que estes poucos dados transmitidos usarão uma grade quantidade de bytes na rede.

A quantidade de dados transmitida numa rede Ethernet em aplicações industriais é tipicamente menor que 100 bytes [46] um dispositivo Ethernet/IP, por exemplo.

Em sistemas de controle rápido com baixo tempo de resposta, o tráfego excessivo de dados na rede pode provocar também problemas em buffer de switchs e impedir outras redes de operarem juntamente com a rede envolvida no controle de processo.

Com a integração das redes na rede Ethernet Industrial, um fator positivo é a possibilidade de haver uma única rede em toda a indústria, desde o chão-de-fábrica até o ambiente de escritório. Para isto, porém, é necessário um controle da rede, para não se usar em demasia a rede e evitar o tráfego excessivo que possa comprometer o tempo de ciclo do processo envolvido.

Diante de tudo o que foi mostrado, pode-se observar que o controle de tráfego para uma quantidade pequena de dados como é o caso da Ethernet na indústria apresentará bons resultados.

3.3. Tempo de processamento

De acordo com [47] e [1] o tempo típico de varredura de um CLP é de 1 a 3 milissegundos e também em [1] o tempo de resposta de sensores é de 1 a 10 milissegundos e no caso de sensores de temperatura o tempo de atualização pode chegar a alguns segundos.

O CLP tem cartões em seu gabinete. No caso do uso de fieldbus, há cartões de rede no gabinete do CLP que na verdade é uma extensão dos seus I/Os. Em um mesmo CLP pode haver um ou mais cartões de rede dependendo da necessidade de pontes de leitura e escrita envolvidas na aplicação.

Estes cartões de rede são responsáveis por receber e enviar dados para os dispositivos na rede. Além disto, eles também trocam dados com a CPU do CLP atualizando os escravos com os dados de saída recebidos da CPU após ou durante a execução do programa de controle.

Os fieldbuses existentes atualmente executam a varredura dos seus escravos em alguns milissegundos. Com a Ethernet a varredura da rede pode ser feita em alguns microssegundos.

Como o tempo de processamento é muito superior ao tempo de varredura da rede Ethernet Industrial, se não houver um correto sincronismo e temporização entre a varredura da rede e o processamento da CPU pode-se ter situações desfavoráveis.

Uma deles seria: o tempo para a execução do programa da CPU é muito maior que o tempo mínimo exigido no processo.

Outra situação: o tempo entre o final de um ciclo e o início de outro ciclo no cartão de rede Ethernet é muito pequeno. Neste caso, a rede irá requisitar a mesma informação varias vezes antes mesmo do programa de controle ser executado. Outro problema neste caso é que o uso do barramento será extremante alto e desnecessário e assim outras redes que poderiam estar incorporadas a este mesmo barramento não estarão devido ao alto tráfego.

3.4. Varredura do Cartão de Rede

Com a evolução para a Ethernet o conceito de cartão de varredura de rede continua o mesmo. A varredura da rede é muito mais rápida e pode-se também colher dados da CPU para serem enviados para outros aplicativos como um computador em um local remoto ou mesmo para um computador externo à planta ou a um programa supervisório.

Os cartões de rede podem ser lidos assincronamente [47]. Desta forma, um cartão de rede Ethernet transferirá para o CLP os dados de entrada e receberá os dados de saída. Como esta comunicação é assíncrona, não se tem nenhuma certeza do tempo de resposta do programa que está sendo executado na memória da CPU do CLP em relação ao tempo de varredura da rede. Ou seja, a forma de comunicação no barramento do CLP não é otimizada.

Outro problema é que o cartão de rede Ethernet não faz nenhuma diferenciação entre dispositivos de entrada e de saída ao executar a varredura da rede. Se houvesse um

escalonamento que fizesse a leitura dos escravos de entrada, a execução do programa de controle e depois a atualização dos escravos de saída, a rede teria uma melhor desempenho.

4. SOLUÇÃO PROPOSTA

Para se exemplificar o problema de tráfego e propor uma solução possível através do escalonamento de mensagens, foram implementados softwares que simularão as condições de tráfego bem como a análise deste tráfego em uma rede real.

4.1. CLP e Cartão de Rede

Os CLP existentes no mercado trabalham numa rotina cíclica onde os dados de entrada são lidos, a lógica de controle é executada e os dados de saídas são atualizados [47]. Assim, qualquer alteração num valor de dado de entrada corresponderá a uma respectiva ativação ou desativação de um dado de saída previamente estabelecido.

Os cartões de rede não seguem nenhum padrão especifico de varredura. Um exemplo de tipo de varredura é por endereçamento: a comunicação é estabelecida primeiramente com o escravo de menor endereço até o escravo de maior endereço na rede. Assim, os dados de entrada e saída desta rede são transmitidos e recebidos à CPU do CLP.

A proposta deste trabalho será o desenvolvimento de softwares que apresentem as características de um CLP citadas acima e possam simular uma comunicação real apresentando uma melhora no tempo de ciclo. Este sistema pode ser empregado com uma melhor eficácia em aplicações de controle de movimento, por exemplo, ou outros sistemas com baixo tempo de ciclo

4.2. Simulação da rede

Para simular a rede vão ser desenvolvidos softwares que realizem e analisem a comunicação.

Um software irá gerar os arquivos necessários para a execução dos demais. Mais dois outros softwares irão representar o mestre e o escravo com as trocas de dados. O software do mestre terá uma área destinada ao processamento da CPU do CLP.

Para se aproximar mais da arquitetura de um CLP, o software do mestre representa um cartão de rede em um gabinete de CLP. Após a leitura de todos os escravos uma tabela interna de entrada é preenchida, a lógica de controle é executada e uma segunda tabela de saída também é preenchida para que o mestre atualize os escravos que tenham saída, Figura 25.

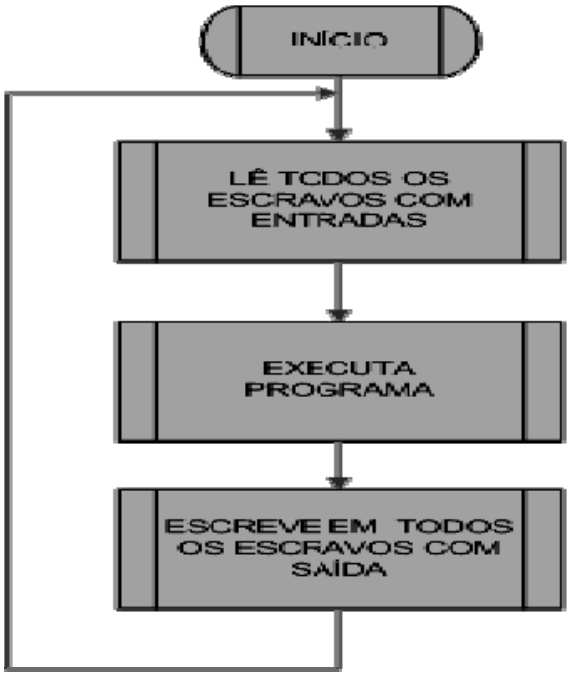

Figura 25 - Fluxograma Geral

O mestre e o escravo só trocam dados, semelhante a qualquer cartão de rede comum. A execução da lógica de controle é feita alterando-se a tabela de saída de acordo com a tabela de entrada, o que pode ser o comportamento de uma CPU de um CLP.

Assim, tenta-se aproximar-se da arquitetura de um CLP com uma CPU e um cartão de rede.

A eventual implementação ou alteração na comunicação e na ordem de comunicação de qualquer CLP não seria possível por terem todos estes códigos proprietários fechados o que justifica esta nova implementação descrita neste trabalho.

Na seqüência será detalhado o funcionamento de cada software.

4.3. Software do mestre

4.3.1. Funções do mestre

A função do mestre é executar a troca de dados com todos os escravos da rede. Ele é responsável por ler todos os escravos, preencher a tabela de entrada com os dados recebidos dos escravos de entrada e também enviar aos escravos de saída os dados existentes na tabela de saída.

Após a comunicação com os escravos com entradas ser executada, o mestre executará um programa de controle previamente estabelecido.

Em seguida, o mestre executa a lógica do controle e comunica-se com os escravos que tenham saídas.

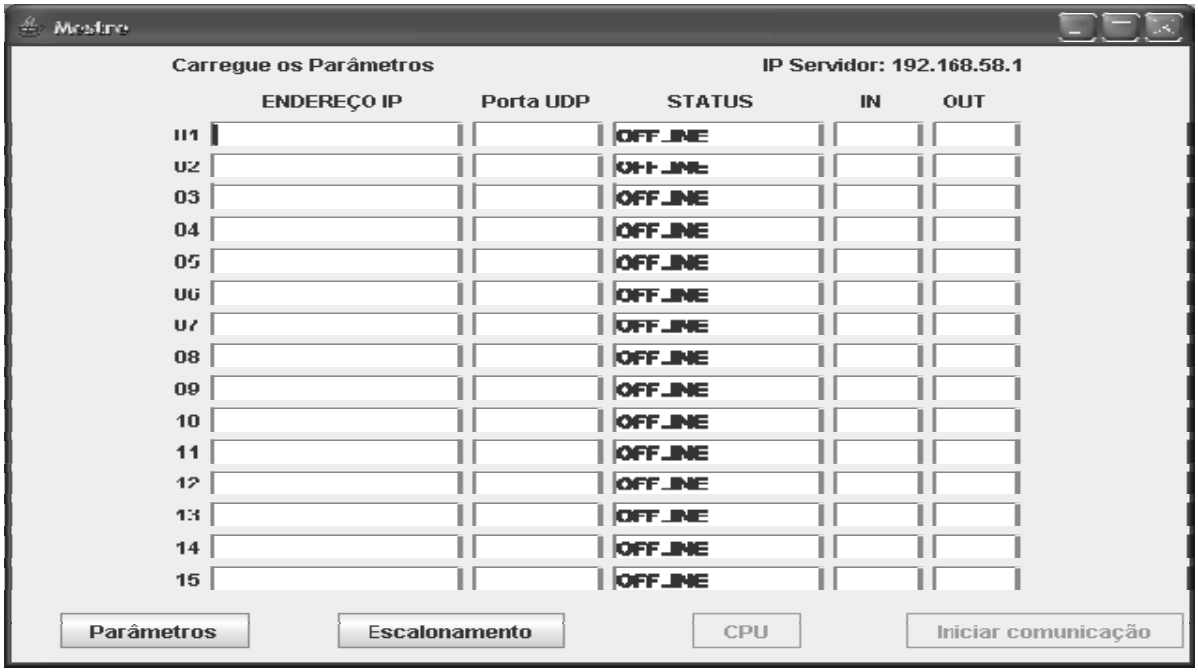

Figura 26 - Interface Principal do Mestre

4.3.2. Descrição do Fluxograma do Mestre

O fluxograma do mestre está representado da Figura 27.

Após inicialização, os parâmetros dos escravos deverão ser carregados ou os escravos deverão ser previamente iniciados, pois o mestre pode detectar a presença de escravos na rede local. Isto é feito na inicialização dos escravos quando é enviada uma mensagem broadcasting na rede identificando ao mestre que o escravo encontra-se no modo “on-line”.

No arquivo de parâmetros há informações sobre endereço de todos os escravos bem como a porta UDP utilizada na comunicação. Carregando os parâmetros, o mestre terá condições de identificar todos os escravos da rede. Na Figura 28 há um exemplo do arquivo com os escravos carregado no mestre.

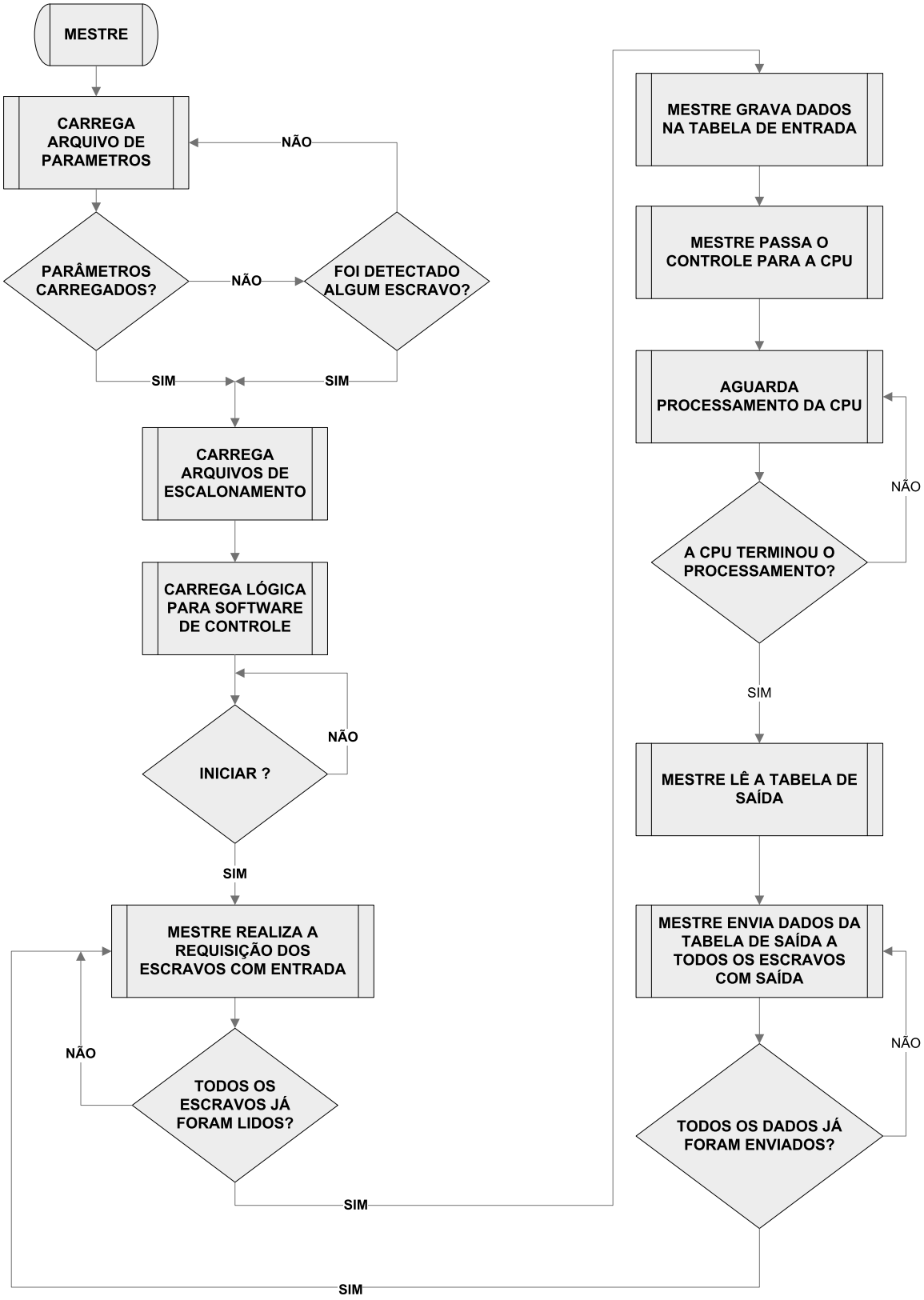

Figura 27 - Fluxograma Mestre

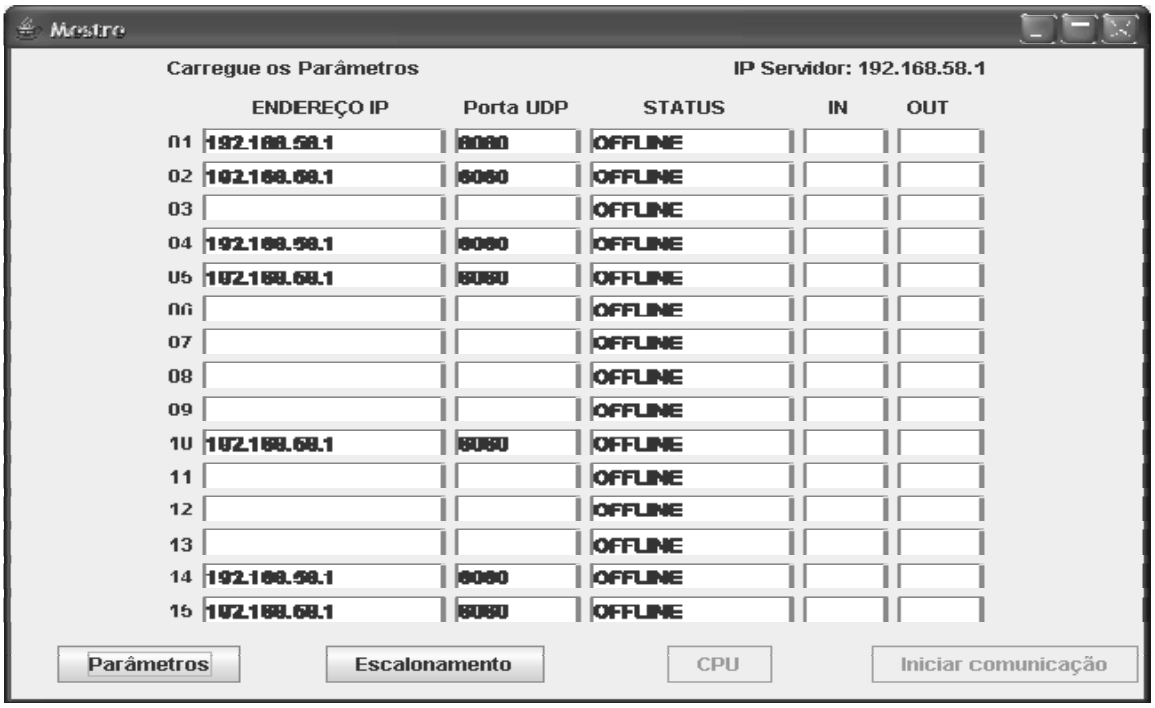

Figura 28 - Mestre com Parâmetros Carregados

Se for usada a detecção automática pelo mestre o botão de parâmetros não estará habilitado conforme visto na Figura 29.

O software do mestre é executado da seguinte forma:

- Após iniciado, o software aguarda os escravos enviarem suas mensagens de "on-line" no barramento. Estas mensagens são enviadas via UDP em broadcasting pelos escravos na rede. Não é necessário para os escravos um primeiro comando do mestre para enviarem a mensagem de identificação.
- Se os parâmetros forem carregados e/ou os escravos forem detectados na rede, pode-se, assim, carregar o arquivo de escalonamento.
- O arquivo de escalonamento será usado para determinar a correta execução da comunicação com os dispositivos na rede. Isto é feito antes da rede entrar em funcionamento.
- Na seqüência, verifica-se se os escravos listados no arquivo de configuração são os mesmos escravos detectados na aquisição na primeira fase do software.
- Após iniciado a comunicação, a primeira etapa é a comunicação com os escravos de entrada de forma a se preencher a tabela de entrada a ser enviada para processamento. Este processo é repetido até todos os escravos de entrada a serem devidamente lidos.

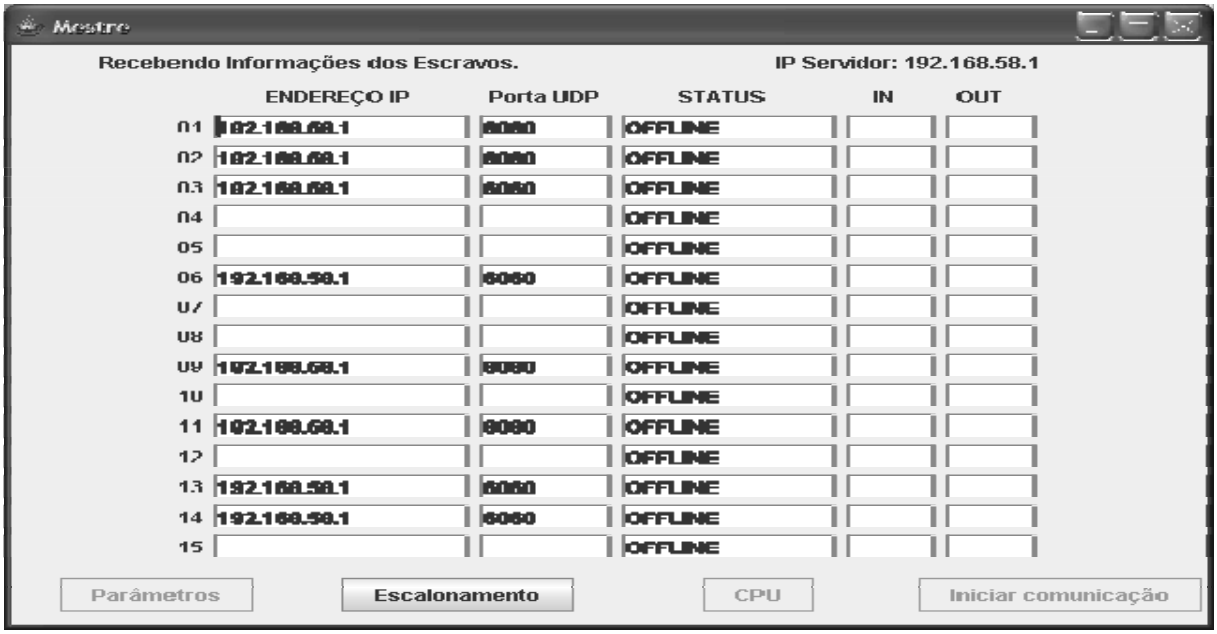

Figura 29 - Detecção Automática dos Escravos

- Finalizada a leitura de todos os escravos de entrada e gravados estes dados na tabela de entrada, o controle é passado para a CPU que irá executar o programa.
- Aguarda-se a completa execução do programa existente na CPU.
- Após a CPU ter executado o programa de controle, a tabela de saída estará com todos os dados atualizados.
- Os escravos de saída recebem, então, os dados da tabela de saída.
- O processo se reinicia novamente com a leitura dos dados de entrada.

O tempo entre a comunicação de um escravo com o mestre e entre a espera do processamento da CPU é controlado pelo arquivo de escalonamento.

4.3.3. Análise do Software do Mestre

A Comunicação será realizada de acordo com o diagrama de interação mostrado na Figura 30.

A comunicação será feita primeiramente com os escravos de entrada que responderão à requisição de dados (“REQ”) do Mestre. Após o termino da leitura dos dados de entrada, a CPU executará o programa e então os dados de saída serão enviados aos respectivos escravos com saídas.

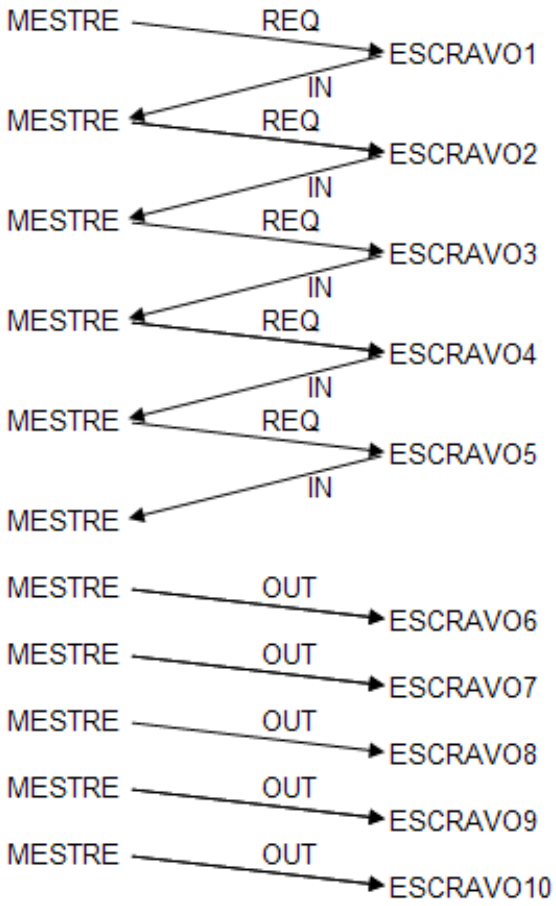

Figura 30 - Diagrama de Interação Mestre x Escravos

Para um melhor entendimento do comportamento da rede é necessário uma ferramenta que forneça as possibilidades de estados deste tipo de sistema.

Em [47], há uma abordagem dos usos da rede de Petri para modelamento de processos industriais. Além disto, a rede de Petri pode ser usada para modelamento de protocolos.

Neste trabalho a rede de Petri será usada de forma a mostrar os estados que o software do mestre apresentará.

Para uma melhor representação e maior funcionalidade, a rede de Petri utilizada será uma rede temporizada. A temporização estará associada aos tempos de transmissão e processamento dos elementos na rede Ethernet.

A temporização da rede será feita nas transições. Assim, somente após um tempo definido a transição será habilitada.

A Figura 31 mostra a rede de Petri representando o fluxograma (Figura 27) do software do mestre, sendo:

- P1: Início do software com aquisição dos escravos disponíveis no barramento;
- P2: Início do escalonamento com os escravos de entrada;

- P3: Transmissão de dados de Entrada;
- P4: Mestre recebeu corretamente os dados do escravo (se houver) e transmite novos dados para o escravo (se houver mais dados a serem transmitidos);
- P5: Escravo recebeu corretamente os dados do Mestre e transmite resposta para este mestre;
- P6: Mensagens enviadas;
- P7: Final da leitura dos escravos de entrada e espera execução da CPU;
- P8: Transmissão de dados de saída;

As transições serão dadas em microssegundos e são:

- T1: 1 microssegundo (tempo de execução do software), início do programa;
- T2: 1 microssegundo (tempo de execução do software), início da comunicação com escravos de entrada;
- T3: 15,4 microssegundos aproximadamente; tempo (teórico) necessário para o mestre transmitir seus dados (72 Bytes UDP) a um escravo qualquer (rede operando em 100 Mbps):

  Tempo TX (ou RX) = 72(bytes)*8(bits)*1/(100*10^6)(tempo de bit) = 5,76 us

  De acordo com [48] é necessário mais um tempo de 9,6 microssegundos entre um frame e outro. Este tempo então deve ser acrescentado ao tempo dos 72 bytes transmitidos, resultando assim em:

  9,6 us+72 Bytes*8 bits*1/100Mbps = 15,4us;

  Para uma rede em 10 Mbps: 9,6 us+72 Bytes*8 bits*1/10Mbps = 67,2us;
- T4: 15,4 microssegundos aproximadamente; tempo necessário para o escravo responder ao mestre (72 Bytes UDP);
- T5: 1 microssegundo (tempo de execução do software), final da comunicação com escravos de entrada;
- T6: Tempo necessário para a CPU executar o programa e início da comunicação com escravos de saída
- T7: 1 microssegundo (tempo de execução do software), final da comunicação com escravos de saída e retorno para iniciar novamente comunicação com escravos de entrada;
- T8: 15,4 microssegundos aproximadamente; tempo necessário para o mestre enviar os dados para o escravo (72 Bytes UDP).

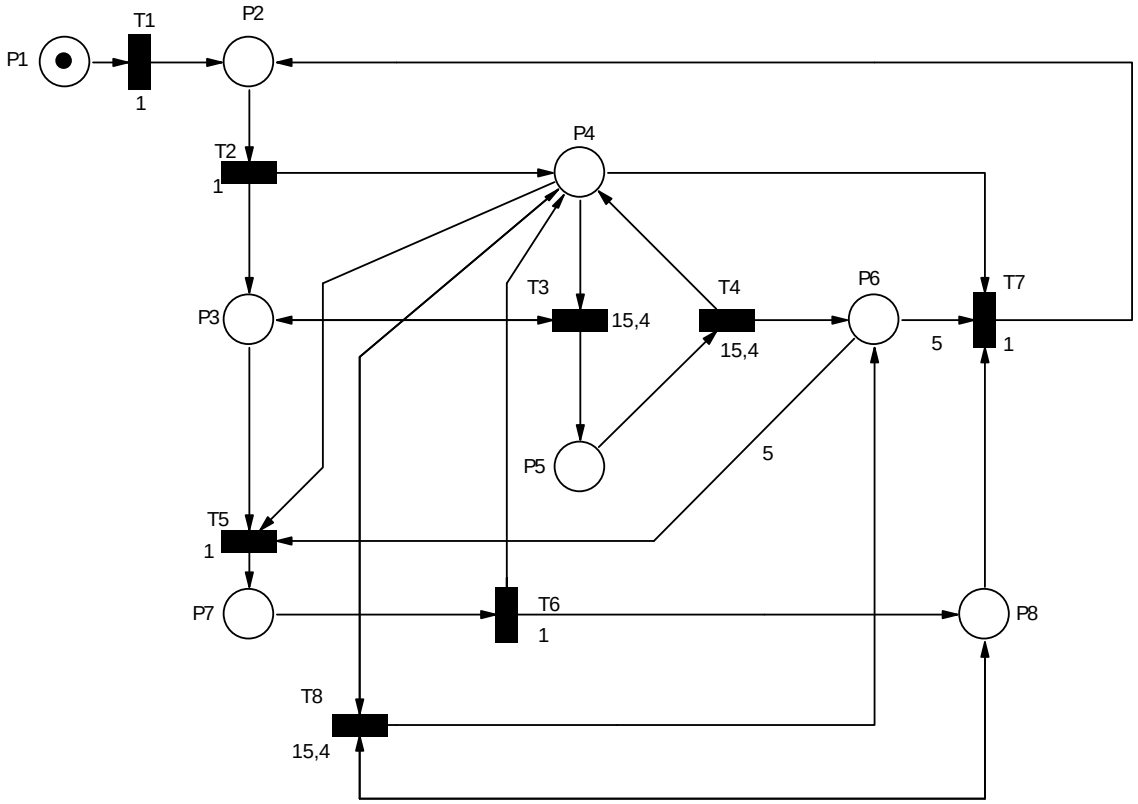

Figura 31 - Rede de Petri do Mestre – Estados

O valor da transição T6 será considerado instantâneo (1 microssegundo), para fim de futuras comparações.

A rede irá representar o software do mestre comunicando-se com cinco escravos de entrada e cinco escravos de saída, o que resultará num tráfego total de 1080 bytes em cada ciclo.

Para a representação da rede de Petri será utilizado o software Visual Object Net++ [49] que é capaz de simular uma rede de Petri temporizada.

Assim, de acordo com [47], fazendo-se a árvore de alcançabilidade da rede de Petri tem-se como resultante a Tabela 3 onde Mx indica os estados possíveis da rede com suas respectivas fichas nas posições P1, P2, P3, P4, P5, P6, P7 e P8.

Utilizando-se o software Visual Object Net++ é possível acompanhar a execução da rede de acordo com o tempo. Pode-se, por exemplo, calcular o tempo que a rede necessita para executar uma varredura (um ciclo na rede). Conforme pode ser visto na Tabela 3, este tempo, para este exemplo, foi de 236 microssegundos.

Mx [P1 P2 P3 P4 P5 P6 P7 P8]	Tempo [us]
M0 [1 0 0 0 0 0 0 0]	0,0
Transição T1	
M1 [0 1 0 0 0 0 0 0]	1,0
Transição T2	
M2 [0 0 1 1 0 0 0 0]	2,0
Transição T3	
M3 [0 0 1 0 1 0 0 0]	17,4
Transição T4	
M4 [0 0 1 1 0 1 0 0]	32,8
Transição T3	
M5 [0 0 1 0 1 1 0 0]	48,2
Transição T4	
M6 [0 0 1 1 0 2 0 0]	63,6
Transição T3	
M7 [0 0 1 0 1 2 0 0]	79,0
Transição T4	
M8 [0 0 1 1 0 3 0 0]	94,4
Transição T3	
M9 [0 0 1 0 1 3 0 0]	109,8
Transição T4	

Mx [P1 P2 P3 P4 P5 P6 P7 P8]	Tempo [us]
M10 [0 0 1 1 0 4 0 0]	125,2
Transição T3	
M11 [0 0 1 0 1 4 0 0]	140,6
Transição T4	
M12 [0 0 1 1 0 5 0 0]	156,0
Transição T5	
M13 [0 0 0 0 0 0 1 0]	157,0
Transição T6	
M14 [0 0 0 1 0 0 0 1]	158,0
Transição T8	
M15 [0 0 0 1 0 1 0 1]	173,4
Transição T8	
M16 [0 0 0 1 0 2 0 1]	188,8
Transição T8	
M17 [0 0 0 1 0 3 0 1]	204,2
Transição T8	
M18 [0 0 0 1 0 4 0 1]	219,6
Transição T8	
M19 [0 0 0 1 0 5 0 1]	235,0
Transição T7	
M1 [0 1 0 0 0 0 0 0]	236,0

Tabela 3 - Árvore de Alcançabilidade

No estado M19 quando a transição T7 é disparada, a rede volta ao estado M1 reiniciando-se o ciclo de leitura da rede.

Após 156 microssegundos a rede de Petri se encontrará na situação mostrada na Figura 32. Na Tabela 3, esta posição seria a posição M12. Neste caso significaria que a rede já se comunicou com todos os cinco dispositivos de entrada.

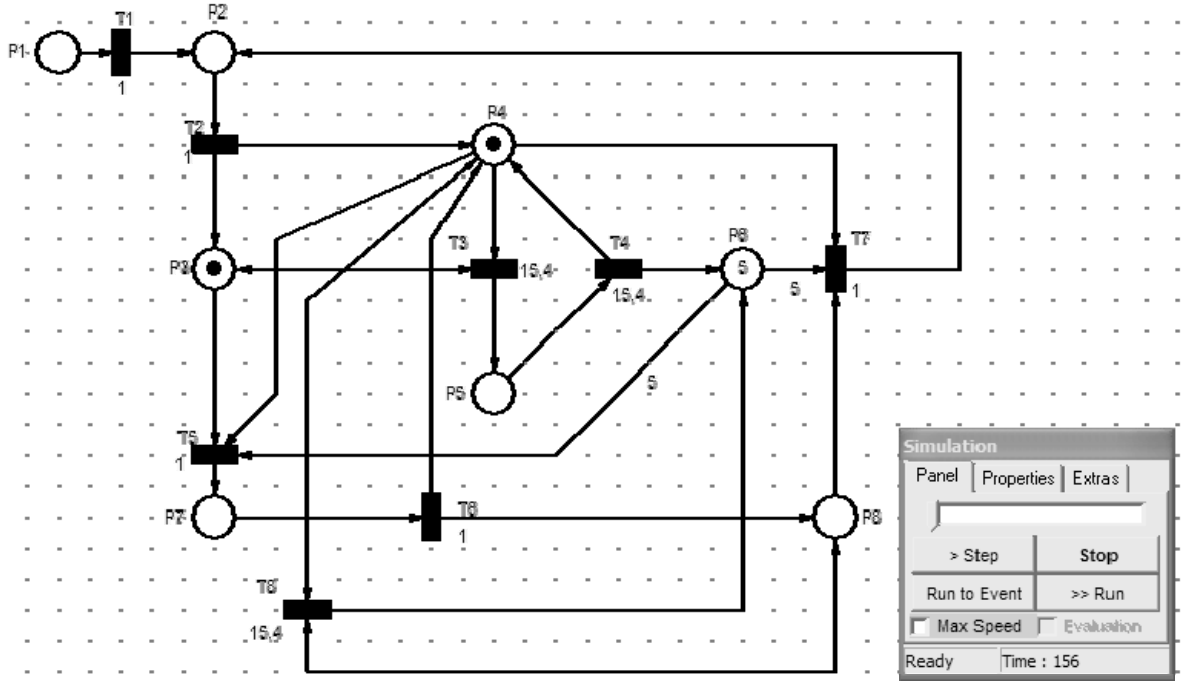

Figura 32 - Rede de Petri: Final da Transmissão aos Escravos com Entradas

Da mesma forma, a Figura 33 indica que o mestre já executou a atualização de todos os escravos de saída da rede. Isto irá acontecer, conforme a árvore de alcançabilidade na Tabela 3, no tempo de 235 microssegundos.

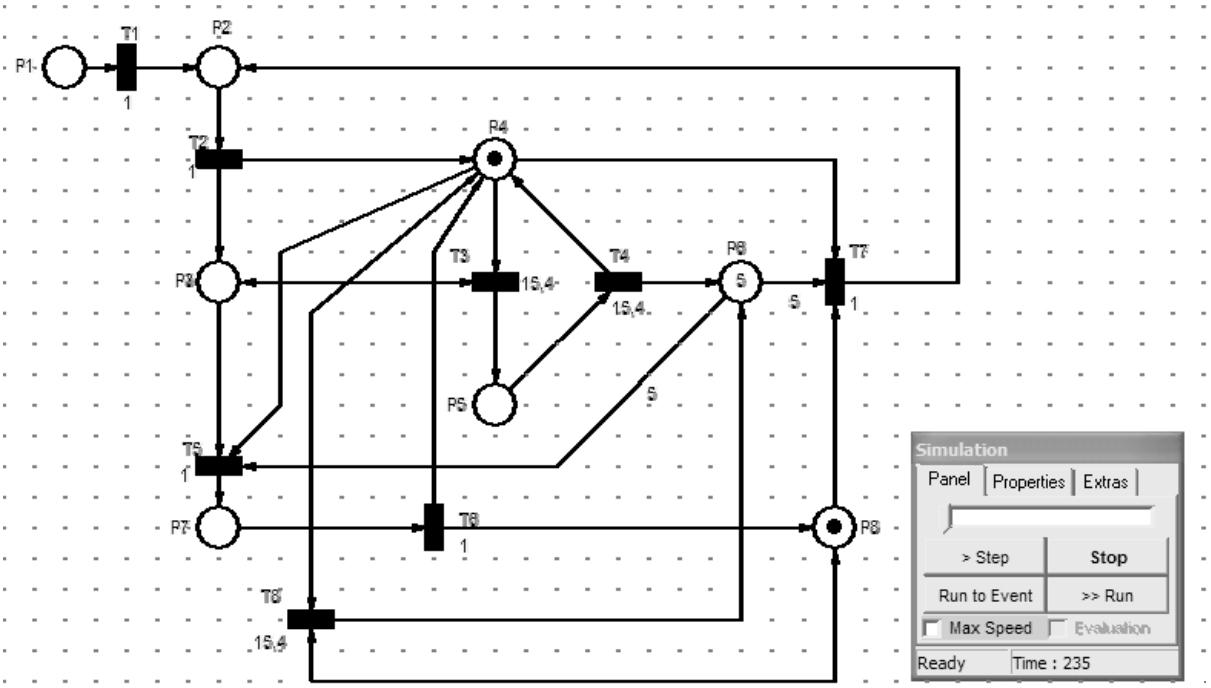

Figura 33 - Rede de Petri: Final da Transmissão aos Escravos com Saídas

Para a análise do software através da árvore de alcançabilidade, Tabela 1, faz-se necessário saber as probabilidade que cada estado terá com o sistema sendo executado continuamente. Para estes cálculos pode ser utilizada a teoria da cadeia de Markov [50].

A árvore de alcançabilidade pode ser associada à cadeia de Markov onde os nós da árvore correspondam aos estados da cadeia de Markov.

O cálculo do vetor de probabilidade [50] de estado é feito pela equação:

$\pi.Q = 0$

Onde:

π é o vetor probabilidade num estado estacionário

Q é a matriz taxa de transição com:

$Q = \frac{1}{\text{Tempo Médio}}$ e

$\sum$ linhas = 0

Através do vetor de probabilidade podem-se descobrir as probabilidades de utilização e não utilização da rede através dos estados em que o sistema possa estar.

Tem-se, então, a Matriz de 25 linhas por 25 colunas.

Para uma primeira condição será considerada a comunicação sem intervalo de tempo entre os ciclos.

De acordo com os tempos na Tabela 3 e sabendo que:

Q ji = taxa de transição de j para i e ∑ linhas = 0, pode-se encontrar:

Q 0 1 = Q 1 2 = 1/1 = 1

Q 2 3 = Q 3 4 = Q 4 5 = Q 5 6 = Q 6 7 = 1/15,4 = 0,065

Q 7 8 = Q 8 9 = Q 9 10 = Q 10 11 = Q 11 12 = 1/15,4 = 0,065

Q 12 13 = Q 13 14 = 1/1 = 1

Q 14 15 = Q 15 16 = Q 16 17 = Q 17 18= Q 18 19= 1/15,4 = 0,065

Q 19 1 =1/1 =1

A matriz transição para esta primeira condição será a Figura 34:

Q =

-1,0	1,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	-1,0	1,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-1,0	1,0	0,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-1,0	1,0	0,0	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065	0,0
0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-0,065	0,065
0,0	1,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	-1,0

Figura 34 - Matriz Taxa de Transição para o Mestre na 1ª Condição

Da matriz acima, resulta o seguinte vetor de probabilidade:

π0	0,000000
π1	0,004260
π2	0,065531
π3	0,065531
π4	0,065531
π5	0,065531
π6	0,065531
π7	0,065531
π8	0,065531
π9	0,065531

π10	0,065531
π11	0,065531
π12	0,004260
π13	0,004260
π14	0,065531
π15	0,065531
π16	0,065531
π17	0,065531
π18	0,065531
π19	0,004260

Tabela 4 - Vetor Probabilidade para a 1ª Condição

A posição πx na Tabela 4, onde x = 0,1,2..24, indica a probabilidade da rede se encontrar em um determinado estado, respectivamente para π 0, π 1, π 2,... π 19 tem-se M0, M1, M2...M19. Por exemplo, a probabilidade da rede se encontrar no estado M15 é π 15 = 4,93%.

A ocupação da rede acontecerá nos instantes de M2 até M11 (comunicação com escravos de entrada) + M14 até M18 (comunicação com escravos de saída).

Assim, tem-se:

Porcentagem de ocupação do barramento = π 2+π 3+... π 11 + π 14+ π 15+.. π 18

Porcentagem de ocupação do barramento = 0, 982961992 ou 98,3%

Porcentagem de não ocupação do barramento = π 0+ π 1+ π 12+ π 13 + π 19

Porcentagem de não ocupação do barramento = 0,017038 ou 1,7%

Pode-se observar que o estado M0 não apresenta nenhuma probabilidade, pois ele será executado somente na inicialização, assim, não haverá outra possibilidade de se chegar neste estado novamente.

Se agora, considerando para uma segunda condição uma comunicação escalonada com os escravos e sincronizada com a CPU pode-se ter a transição T6 com o tempo de processamento da CPU.

Em [47] e também em [1], o tempo médio de processamento da CPU do CLP é dado entre 1 a 3 milissegundos.

Adotando-se dois milissegundos para o tempo de processamento, tem-se a modificação:

Q 13 14=1/2000 = 0,0005

E assim, tem-se a matriz transição da Figura 35:

$$Q=\begin{vmatrix}
-1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & -1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,0005 & 0,0005 & 0, & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 & 0, \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -0,065 & 0,065 \\
0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -1,
\end{vmatrix}$$

Figura 35 - Matriz Taxa de Transição para o Mestre na 2ª Condição

Da matriz acima, resulta o seguinte vetor de probabilidade da Tabela 5:

π0	0,000000	π10	0,065531
π1	0,004260	π11	0,065531
π2	0,065531	π12	0,004260
π3	0,065531	π13	0,004260
π4	0,065531	π14	0,065531
π5	0,065531	π15	0,065531
π6	0,065531	π16	0,065531
π7	0,065531	π17	0,065531
π8	0,065531	π18	0,065531
π9	0,065531	π19	0,004260

Tabela 5 - Vetor Probabilidade para a 2ª Condição

E agora:

Porcentagem de ocupação do barramento = π 2+ π 3+... π 11 + π 14+ π 15+.. π 18

Porcentagem de ocupação do barramento = 10,33%

Porcentagem de não ocupação do barramento = π 0+ π 1+ π 12+ π 13 + π 19

Porcentagem de não ocupação do barramento = 89,67%

A principal causa desta diminuição é a probabilidade no estado M13 ter aumentado, pois quando maior o tempo de processamento da CPU, mais a rede ficará ociosa.

Por outro lado, se houver um tempo muito grande dedicado ao processamento da CPU, o tempo de resposta da rede irá diminuir e o processo envolvido poderá ser prejudicado. Assim, deve-se ter a otimização deste tempo de forma a se ter a melhor eficiência da CPU, da rede e consequentemente uma melhor eficiência no controle do processo.

Desta forma, pode-se notar que um tempo maior de processamento gera um tráfego maior na rede.

4.3.4. Parâmetros do Mestre

Conforme mostrado na Figura 28, a interface do mestre tem alguns arquivos a serem carregados antes do início da comunicação.

4.3.4.1. Parâmetros de Configuração de todos os Escravos

O arquivo de configuração dos escravos é um arquivo do tipo “.txt” que deve conter as informações de todos os escravos na rede.

Exemplo deste arquivo para quatro escravos:

01 10.10.3.61 5050 6060

02 10.10.3.62 5050 6060

04 10.10.3.61 5050 6060

13 10.10.3.73 5050 6060

Cada linha tem as informações de um escravo na seguinte seqüência:

- "01" = número do escravo
- "10.10.3.61" = endereço IP do escravo
- "6060" = Porta UDP utilizada

Este arquivo pode ou não ser carregado. Se for carregado previamente, os escravos que se comunicarão com o mestre deverão ser os configurados no arquivo e carregados no software. A outra opção seria iniciar primeiramente a comunicação dos escravos na rede de forma que o Mestre possa identificá-los e criar automaticamente a configuração da rede.

4.3.4.2. Parâmetros de Escalonamento para a Comunicação

Os parâmetros de escalonamento também são feitos em um arquivo do tipo ".txt". A forma como se apresentam é a seguinte:

0000 01

0010 02

0020 03

0030 CPU

1530 04

1540 05

1550 06

Em cada linha é colocado o tempo (em microssegundos) com o endereço do escravo correspondente. Por exemplo, de acordo com o texto acima, no instante 0020 microssegundos o mestre deve se comunicar com o escravo 03. O tempo é dado em microssegundos devido à escala de tempo em que ocorre a comunicação.

Os primeiros escravos, antes da "CPU" devem ser os escravos de entrada e os depois da CPU, os escravos de saída.

Do exemplo acima, no instante inicial zero, o mestre irá se comunicar com o escravo um ("1") lendo suas entradas, após isto, no instante 10 microssegundos a comunicação deve ser feita com o escravo dois ("2"), em 20 microssegundos com o escravo três ("3"). Depois de ler todos os escravos de entrada o mestre deve passar o controle para a CPU executar o programa de controle. O tempo de processamento da CPU neste exemplo foi de 1,5

milissegundos ou 1500 microssegundos. Após o programa de controle ter sido executado, o mestre retorna a ocupar o barramento e agora atualiza todos os escravos que tenham saída.

Quando chega ao final do arquivo o ciclo é novamente reiniciado.

4.3.4.3. Programa de Controle para a CPU

O software de controle é feito da seguinte forma:

- O primeiro termo refere-se ao primeiro escravo de saída, a assim a segunda linha ao segundo escravo e a terceira linha representa o terceiro escravo.

Exemplo:

01 04 +

06 07 –

01 02 +

Por uma questão de teste, as operações válidas entre escravos é a soma (“+”) e a subtração (“-“).

No exemplo acima, o primeiro escravo de saída recebe a valor de entrada do escravo 01 mais o valor de entrada do escravo 04 (“01 04 +”).

4.3.5. Comunicação UDP no Mestre

Na inicialização do software para a detecção de escravos é usado também o UDP. Assim, o mestre é capaz de localizar todos os escravos que estarão se comunicando com ele naquele dado instante.

O mestre usa o protocolo UDP para requisitar os dados de entrada dos escravos e também para enviar dados de saídas para os escravos.

4.4. Software do Escravo

4.4.1. Fluxograma do Escravo

A Figura 36 mostra o fluxograma do software do escravo.

Depois de inicializado, o software irá carregar seus parâmetros que conterá o endereço, a porta UDP e as variáveis de entrada e de saída.

Na seqüência o escravo entrará em uma rotina de broadcasting, onde ele irá divulgar seus dados para um eventual mestre presente na rede em que se encontra. Se não receber a

confirmação do mestre, ocorre um time-out e o software é finalizado aguardando uma nova solicitação de início pelo usuário.

Se o escravo receber a confirmação do mestre, ele entra na rotina de troca de dados.

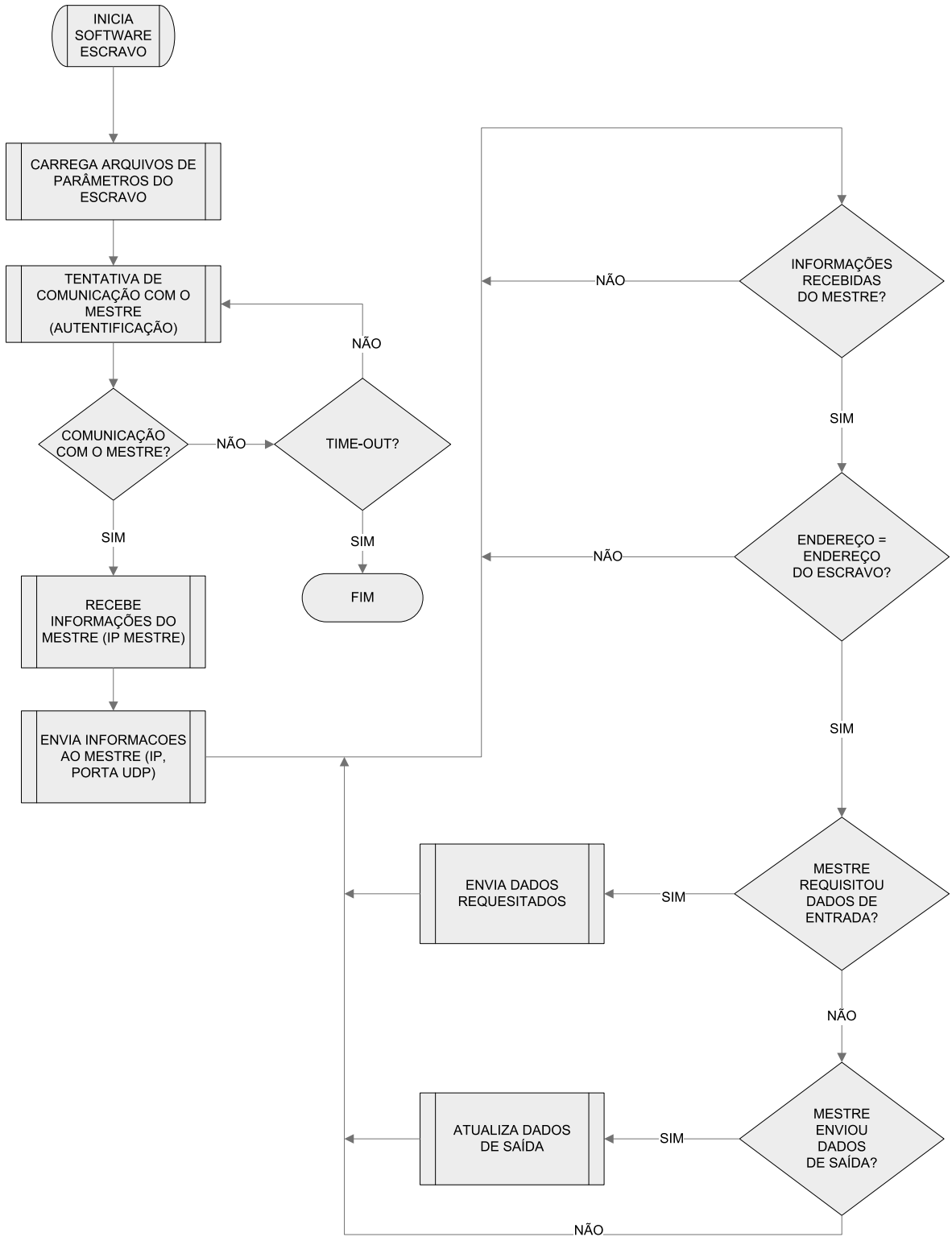

Figura 36 - Fluxograma do Escravo

O escravo irá executar esta troca de dados via UDP até que o usuário solicite o termino da comunicação.

Se o escravo for um escravo de entrada, o mestre irá solicitar o envio das informações de entrada e sendo de saída o mestre enviará os dados de saída. Pode haver escravos que sejam de entrada e saída. Assim, na primeira varredura o mestre solicitará a entrada e na próxima, depois de executar o software da CPU, ele enviará os dados das saídas já atualizados.

4.4.2. Arquivo de Configuração dos Escravos

Da mesma forma como no mestre, há também um arquivo de configuração para cada escravo.

O arquivo ".txt" terá o seguinte formato:

01 192.168.58.1 5050 01

O escravo terá a linha que conterá suas informações:

- 01 = número do escravo
- 192.168.58.1 = endereço IP do escravo
- 5050 = Porta UDP utilizada
- 01 = Valor inicial do dado de entrada

A Figura 37 mostra a interface do escravo.

Figura 37 - Escravo Inicializado

Após a inicialização do escravo é necessário carregar-se o arquivo de parâmetros. Pode-se notar que na Figura 37 não há nenhuma informação disponível, somente o IP local do computador. Na Figura 38, é mostrada a tela onde se carrega os parâmetros do escravo.

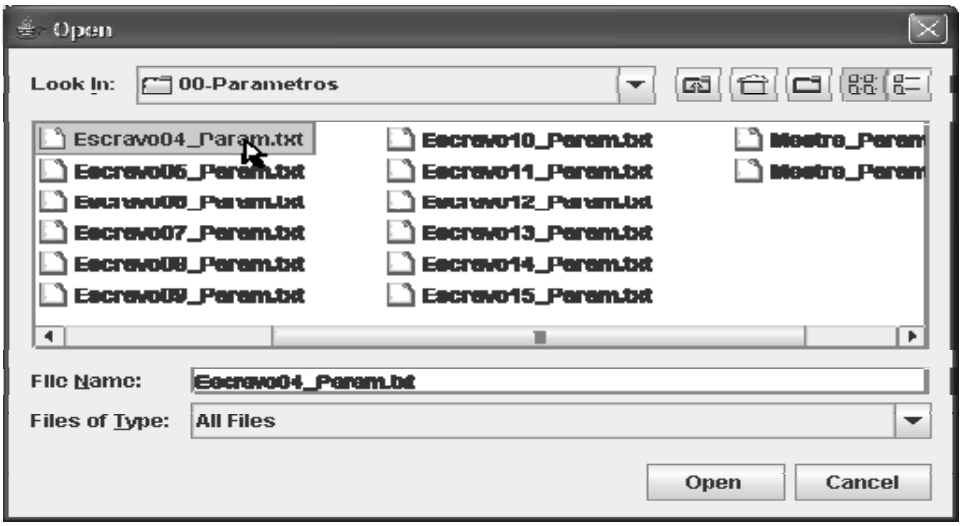

Figura 38 – Tela de carga do arquivo com os Parâmetros do Escravo

Após se carregar os parâmetros de configuração do escravo, a interface principal irá mostrar as informações, conforme Figura 39. É possível se saber o IP do mestre através do campo "IP Servidor" que representará o ip do mestre daquele escravo.

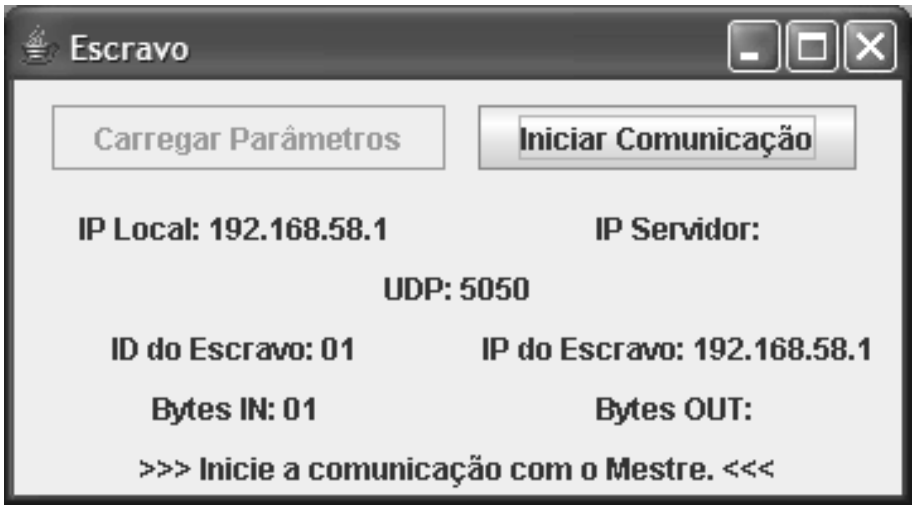

Figura 39 - Parâmetros Carregados Corretamente no Escravo

Depois de carregado o arquivo de parâmetros, o campo "Iniciar comunicação" estará liberado para ser acionado. Se houver um erro no arquivo, como o IP errado, por exemplo, a comunicação não poderá ser iniciada e será mostrada uma mensagem de erro, conforme Figura 40.

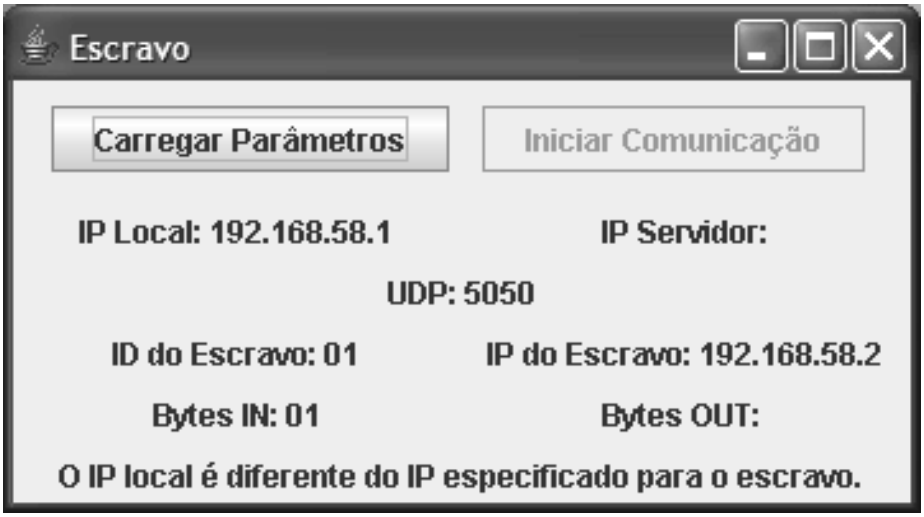

Figura 40 - Arquivo de Parâmetro com Erro

Quando se inicia a comunicação com o mestre, o escravo envia uma mensagem para que o mestre o identifique. A Figura 41 mostra a interface do escravo após toda a configuração estar completa.

Figura 41 - Dados do Escravo enviados para o Mestre

Para a análise da comunicação na rede Ethernet será utilizado um software de monitoramento da rede. A escolha foi feita pelo Ethereal [51] por ser um software que fornece uma maior quantidade de informações dos dados capturados na rede Ethernet.

Com o Ethereal é possível analisar-se a comunicação entre escravo e mestre. A Figura 42 mostra o frame do escravo capturado no Ethereal.

Conforme Figura 42, pode-se ver o frame Ethernet com 50 bytes encapsulados no protocolo UDP.

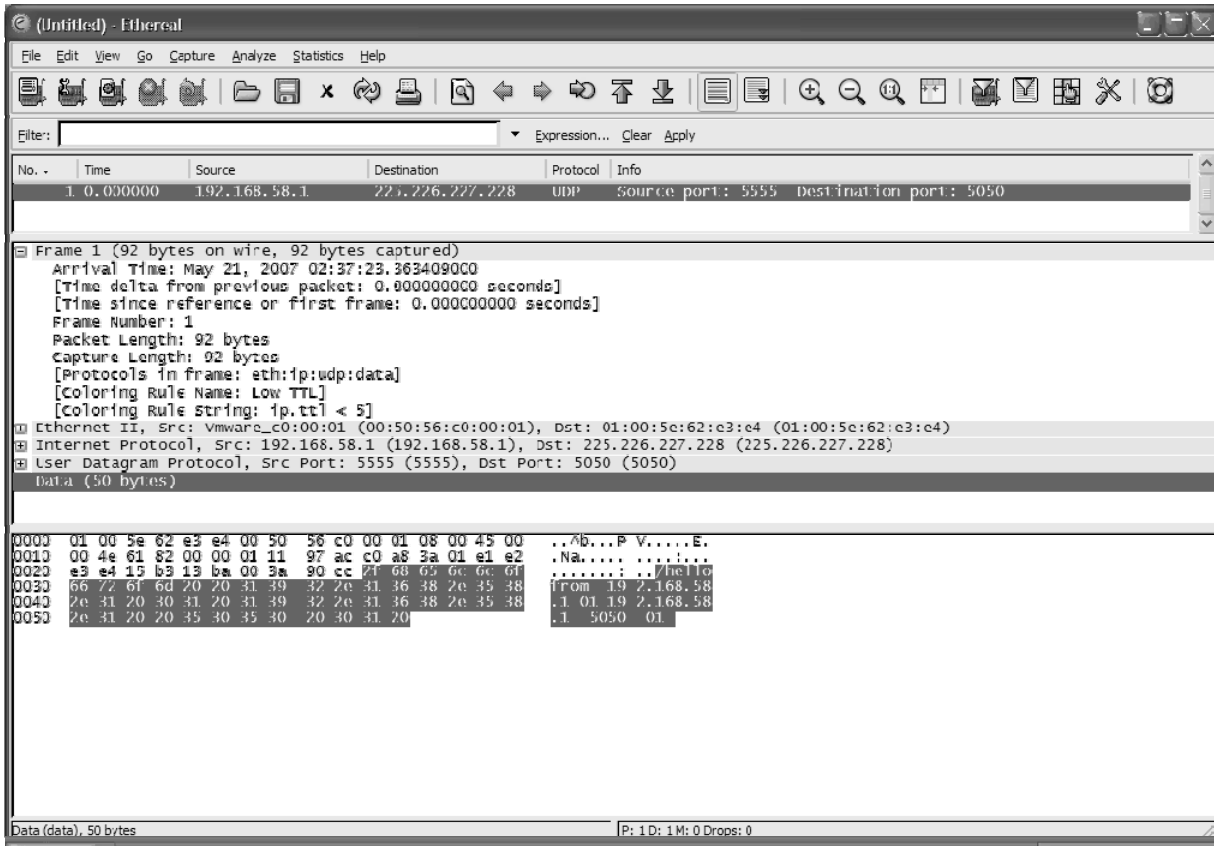

Figura 42 - Anúncio do Escravo 04 Capturado no Ethereal

Nestes 50 bytes está a camada de aplicação que neste caso somente enviou a mensagem:

"/hello from 192.168.58.1 01 192.168.58.1 5050 01"

Nesta mensagem está o endereço IP do escravo "192.168.58.1", o endereço (número do escravo) "01", a porta UDP "5050" e o valor inicial "01" da variável de entrada.

Da mesma forma, se houvesse mais escravos, mais mensagens seriam geradas nesta mesma sintaxe.

A função do escravo é enviar e receber dados de acordo com a solicitação do mestre. Os dados presentes nas entradas do escravo serão transmitidos ao mestre e sua saída será atualizada com os dados recebidos do mestre.

4.5. Comunicação mestre-escravo

Na Figura 43, pode-se notar que quando é carregado o arquivo de escalonamento o mestre enviará uma mensagem para todos os escravos informando ser ele o mestre para estes escravos.

A mensagem "iam server 10.1.1.22" indica o endereço IP do mestre: 10.1.1.22.

Após isto, com o escalonamento e o software da CPU carregados, a comunicação poderá ser iniciada.

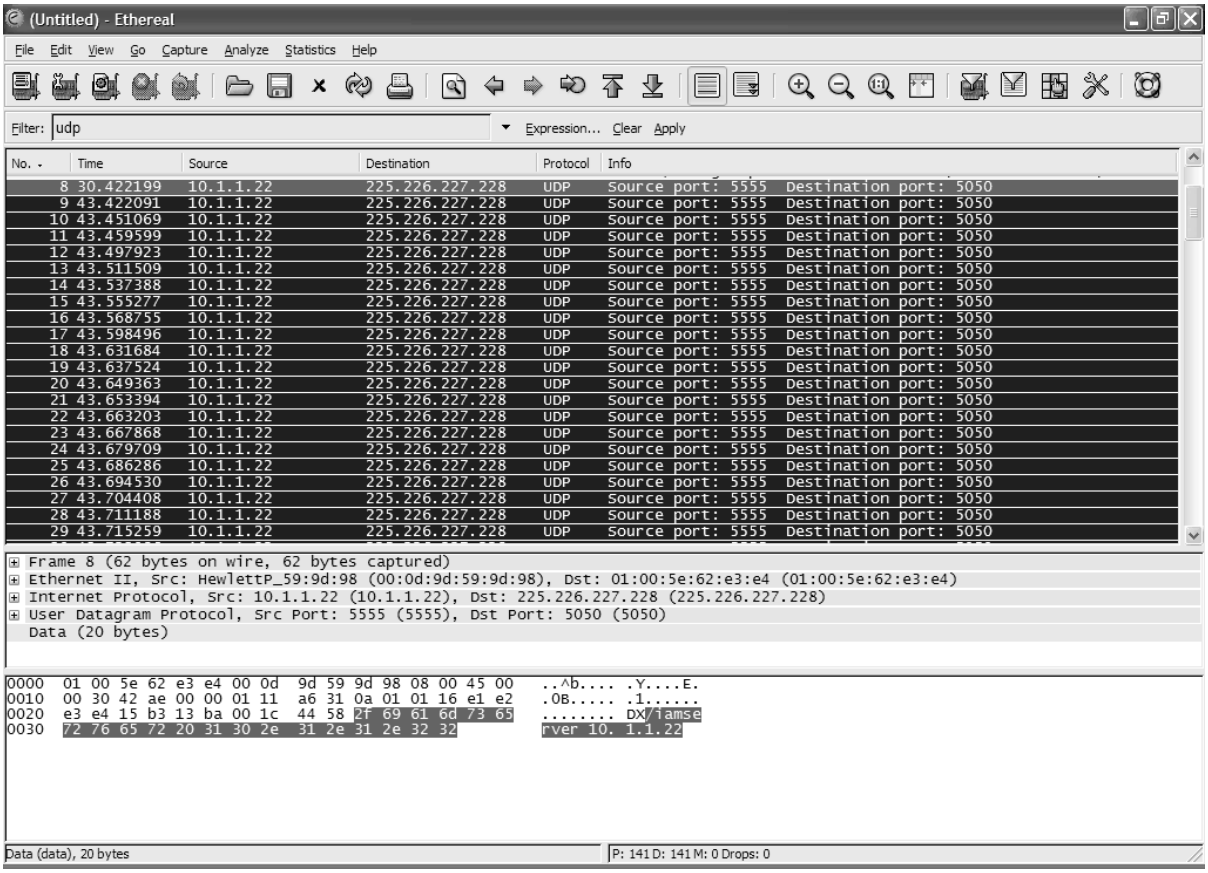

Figura 43 - Mensagem do Mestre aos Escravos

Por exemplo, considerando a seguinte configuração dos escravos:

“01 10.1.1.22 5050

02 10.1.1.22 5050

03 10.1.1.22 5050

04 10.1.1.22 5050

05 10.1.1.22 5050

06 10.1.1.22 5050”

A configuração do escalonamento poderia ser:

“0000 01

0010 02

0020 03

0030 CPU

1530 04

1540 05

1550 06”

E o software de controle:

“01 03 +

02 01 -

02 03 +”

A comunicação acontecerá na seguinte seqüência:

Frame 01: /iam server 10.1.1.22

Frame 02: /hello from 10.1.1.22 01 10.1.1.22 5050 01

Frame 03: /hello from 10.1.1.22 02 10.1.1.22 5050 01

Frame 04: /hello from 10.1.1.22 03 10.1.1.22 5050 02

Frame 05: /hello from 10.1.1.22 04 10.1.1.22 5050 03

Frame 06: /hello from 10.1.1.22 05 10.1.1.22 5050 04

Frame 07: /hello from 10.1.1.22 06 10.1.1.22 5050 05

Frame 08: senddata 01 ; Mestre pede dados de Escravo 1

Frame 09: inputdata 1 01 ; Escravo 1 responde, dado = 1

Frame 10: senddata 02 ; Mestre pede dados de Escravo 2

Frame 11: inputdata 1 02 ; Escravo 2 responde, dado = 1

Frame 12: senddata 03 ; Mestre pede dados de Escravo 3

Frame 13: inputdata 1 03 ; Escravo 3 responde, dado = 1

... ; Tempo para executar o programa:

... ; Esc4 = Esc1+Esc3 => Esc4=1+1=2

... ; Esc5 = Esc2-Esc1 => Esc5=1-1=0

... ; Esc6 = Esc2+Esc3 => Esc6=1+1=2

Frame 14: /rcvdata 04 2 ; Esc4=2

Frame 15: /rcvdata 05 -1 ; Esc5=0

Frame 16: /rcvdata 06 2 ; Esc6=2

Frame 17: senddata 01

Frame 18: inputdata 2 01 ; Escravo 1 responde, dado = 2

Frame 19: senddata 02

Frame 20: inputdata 2 02 ; Escravo 2 responde, dado = 2

Frame 21: senddata 03

Frame 22: inputdata 2 03 ; Escravo 3 responde, dado = 2

Frame 23: /rcvdata 04 4 ; Executa: Esc4 = Esc1+Esc3 => Esc4=2+2=4

Frame 24: /rcvdata 05 0 ; Executa: Esc5 = Esc2-Esc1 => Esc5=2-2=0

Frame 25: /rcvdata 06 4 ; Executa: Esc6 = Esc2+Esc3 => Esc4=2+2=4

Estes frames acima foram retirados do Ethereal.

4.6. Software Escalonador

O software responsável por gerar o escalonamento da comunicação e a geração dos arquivos é o software escalonador e terá a interface mostrada na Figura 44.

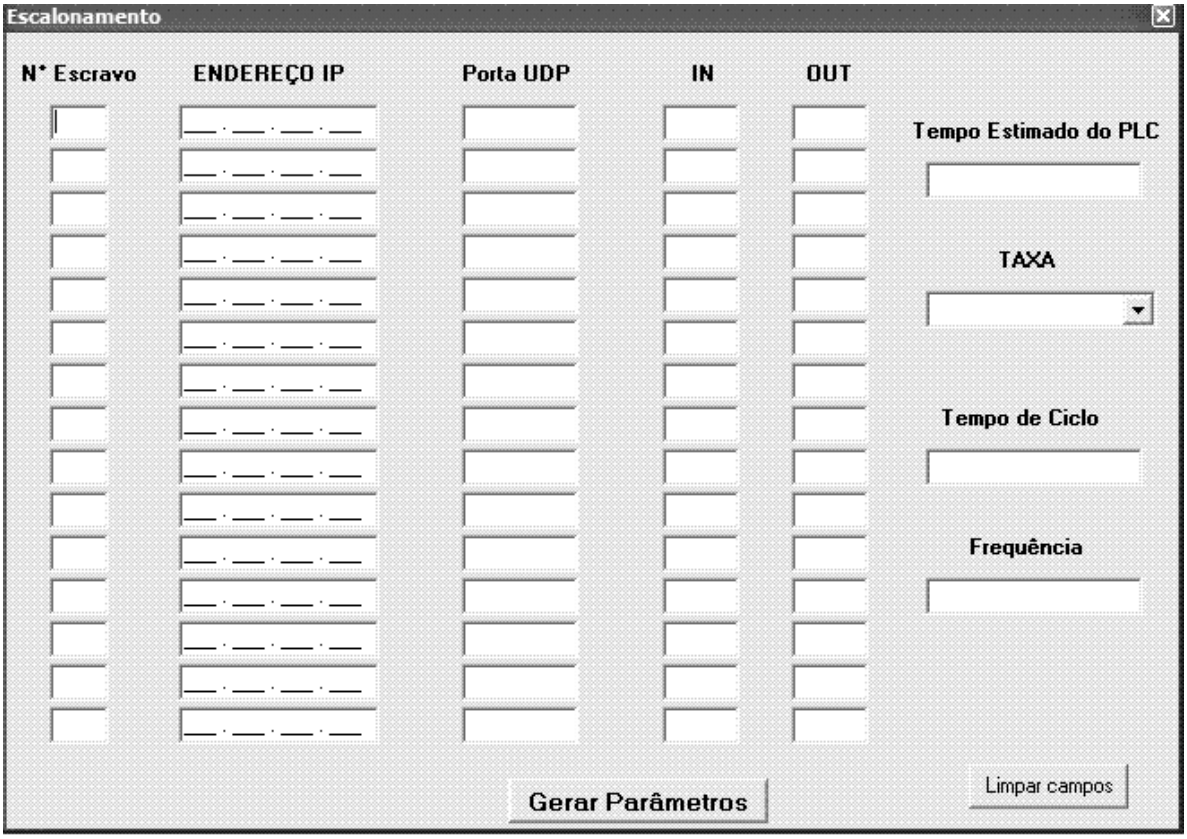

Figura 44 - Interface do Software Escalonador

O software escalonador irá predeterminar a comunicação de forma a ocupar menos possível o barramento e também estabelecerá a comunicação com os escravos com entrada e depois com os escravos com saída. Assim, dispositivos de entrada serão lidos primeiramente, o programa será executado e os dispositivos de saída serão atualizados.

O software irá gerar um arquivo que será compilado no mestre de forma a seguir exatamente a comunicação pré-definida, por exemplo:

0000 01 ; no instante 1 microssegundos é feita a comunicação com o escravo 01
; leitura das entradas.

0010 02 ; no instante 10 microssegundos é feita a comunicação com o escravo 02
; leitura das entradas.

0020 03 ; no instante 20 microssegundos é feita a comunicação com o escravo 03
; leitura das entradas.

0030 CPU ; no instante 30 microssegundos ocorrerá a execução do software da CPU
; para este caso 1500 microssegundos, ou 1,5 milissegundos).
1530 04 ; no instante 1530 microssegundos é feita a comunicação com o escravo 04
; atualização das saídas.
1540 05 ; no instante 1540 microssegundos é feita a comunicação com o escravo 05
; atualização das saídas.

Neste software, serão gerados:

- Escalonamento da comunicação com os escravos envolvidos e a temporização da CPU;
- Parâmetros de inicialização do mestre que contém todos os escravos com seus respectivos endereços, porta UDP e a quantidade de bytes de entrada e saída;
- Parâmetros de cada escravo estabelecido na tela. Se houver 10 escravos, serão gerados 10 arquivos com seus respectivos endereços, porta e quantidade de bytes.

Estes arquivos podem ser gerados através do botão “Gerar Arquivos”.

Com todos os dados preenchidos é possível se verificar o tempo teórico de todo o ciclo que a rede poderá desenvolver.

Estes dados são importantes porque através deles é possível verificar se o sistema está atendendo às necessidades do processo que esta sendo controlado.

Toda a temporização gerada no arquivo de escalonamento é teórica, pois ela leva em consideração que o sistema será capaz de processar instantaneamente os dados a transmitidos e recebidos o que não necessariamente será possível, principalmente em taxas de transmissão muito altas. Todas estas informações serão analisadas posteriormente quando da análise dos resultados.

4.6.1. Fluxograma do Software Escalonador

A Figura 45 mostra o fluxograma do software escalonador.

Na sub-rotina de geração das tabelas de parâmetros para o mestre, é gerado um arquivo “Mestre_parametros.txt”, que terá todas as informações contidas no campo de cada escravo. Este arquivo estará no padrão aceito pelo software do mestre.

Da mesma forma serão gerados todos os softwares dos escravos já no padrão para serem carregados no software dos escravos.

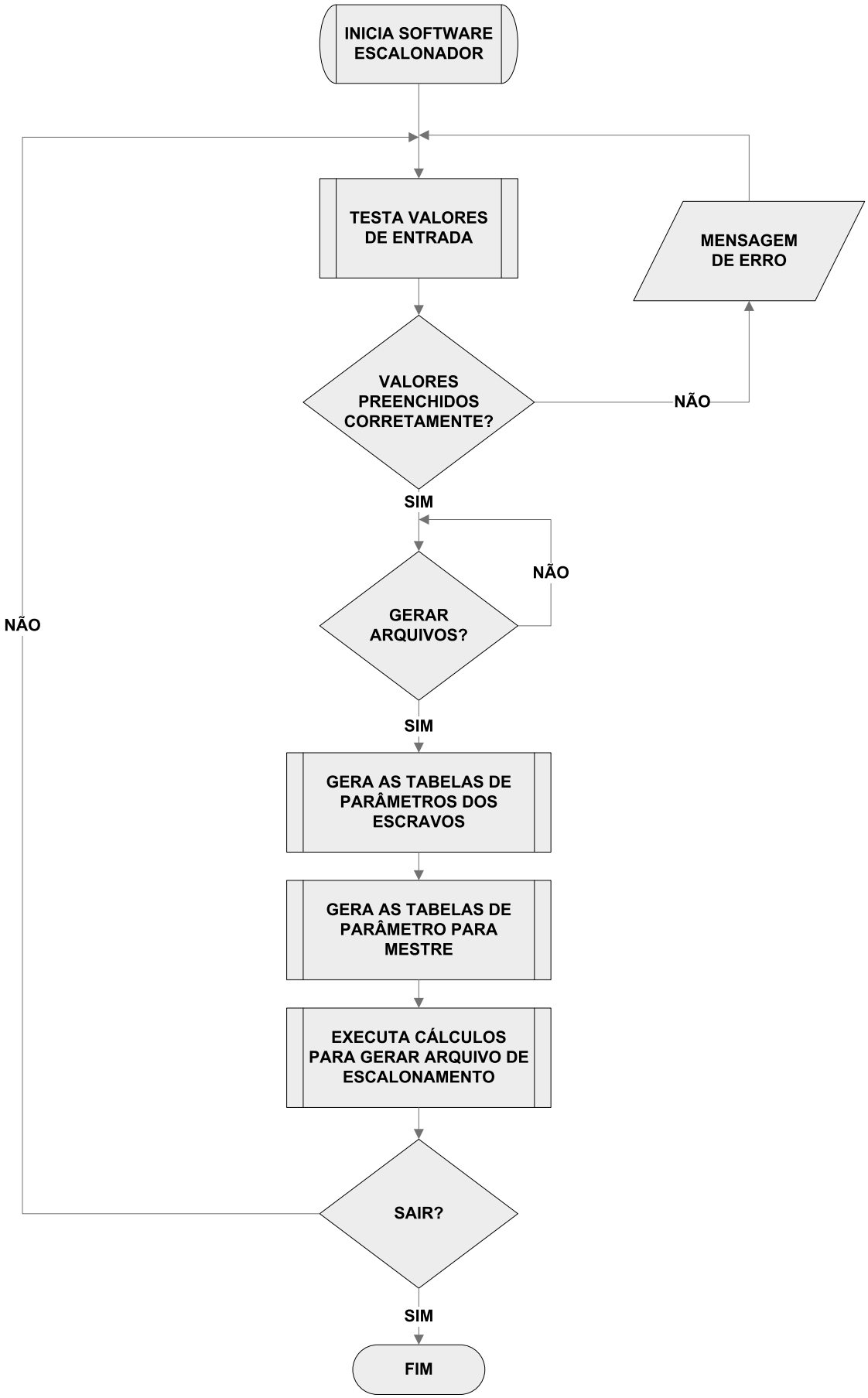

Figura 45 - Fluxograma do Software Escalonador

Na sub-rotina de geração do arquivo de escalonamento, os seguintes passos são executados:

- Procura todos os escravos de entrada;
- Procura todos os escravos de saída;
- Calcula tempo de bit;
- Cálculo do tempo necessário para transmissão para cada escravo de entrada e de saída;
- Acréscimo do tempo de processamento da CPU;
- Somatório de tempo para se achar o tempo do ciclo;

O tempo de bit é calculado pela fórmula:

Tbit= 1 / (taxa)

Assim, tem-se o tempo de bit de:

Tbit = 1/ (10 Mega bits por segundo) = 100 nanossegundos

Tbit = 1/ (100 Mega bits por segundo) = 10 nanossegundos

Tbit = 1/ (1000 Mega bits por segundo) = 1 nanossegundos

Mesmo que um escravo só tenha dados de entrada, na implementação feita, será necessário enviar uma mensagem a este escravo para que ele forneça os seus dados. Como esta mensagem é pequena, seu tamanho terá o mínimo de 72 bytes.

Entre um frame e outro, de acordo com [48], deve haver um tempo de 9,6 us.

Como a comunicação foi implementada em UDP, a quantidade de dados mínima para a transmissão será:

- Total = 72 bytes
- Frame IEEE 802.3 usa 8+6+6+2+4 = 26 bytes
- IP usa de cabeçalho = 20 bytes
- O UDP usa de cabeçalho = 8 bytes

Assim:

- Disponível = 72 – 26 – 20 –8 = 18 bytes

O cálculo do tempo de transmissão será mostrado na seqüência.

Cálculo do Tempo para Escravo com Entrada:

- Se quantidade de dado de entrada =< 18 Bytes

  Tempo = 72*8*Tbit (pergunta ao escravo) + 9,6 us + 72*8*Tbit (resposta com os dados de entrada) + 9,6 us

- Se quantidade de dado de entrada > 18

  Tempo = 72*8*Tbit (pergunta ao escravo) + 9,6 us + [54+ (Quantidade de Dados de Entrada)] *8*Tbit (resposta com os dados de entrada) + 9,6 us

Cálculo do Tempo para Escravo com Saída:

- Se quantidade de dado de saída =< 18 Bytes

  Tempo = 72*8*Tbit (resposta com os dados de entrada) + 9,6 us

- Se quantidade de dado de entrada > 18

  Tempo = [54+(Quantidade de Dados de Entrada)]*8*Tbit (resposta com os dados de entrada) + 9,6 us

Se o escravo tiver dados de entrada e saída, os dois cálculos deverão ser feitos.

Assim será estabelecido o momento exato em que o mestre deverá executar a comunicação com cada escravo, iniciar o programa de controle e o momento em que o programa de controle deva ser executado. Sabendo-se que o programa de controle foi executado, pode-se iniciar a comunicação com os escravos de saídas, pois seus dados de saída estão atualizados.

Nos cartões de redes convencionais não há como se determinar a forma que a comunicação acontece e nem em que momento se tem os dados atualizados. O barramento de comunicação dos CLPs com seus cartões seja de entrada ou saída, de rede ou outro qualquer, não apresenta informações do seu tempo de varredura e de comunicação com a CPU.

Geralmente usa-se um valor alto de interscan (tempo entre a varredura de um ciclo e de outro) de forma que a CPU já tenha executado o programa de controle e esteja de acordo com os requisitos da aplicação. Ou, por outro lado, estabelecendo-se um valor baixo de interscan não há como se precisar o momento em que a lógica de controle na CPU do CLP foi executada. Neste último caso, a rede será sobrecarregada com várias varreduras desnecessárias. O ideal é o tempo de varredura ser a freqüência de amostragem que a aplicação necessite.

Como um exemplo para uma aplicação pode-se ter:

- Escravo 1 - 20 bytes de entrada e 00 bytes de saída
- Escravo 2 - 00 bytes de entrada e 50 bytes de saída
- Escravo 3 - 04 bytes de entrada e 00 bytes de saída
- Escravo 4 - 00 bytes de entrada e 10 bytes de saída
- Rede Ethernet a 10Mbps
- Tempo de processamento da CPU: 1 ms

Deve-se encontrar:

O tempo de bit será de: Tbit = 1/(10 000 000) = 100 ns

Tempo para transmissão de cada escravo:

- Escravo 1, Entrada:

  Tempo1 = 72*8*Tbit + 9,6us + 74*8*Tbit + 9,6 us

  Tempo1 = 136 us

- Escravo 2, Saída:

  Tempo2 = 104*8*Tbit + 9,6 us

  Tempo2 = 92,8 us

- Escravo 3, Entrada:

  Tempo3 = 72*8*Tbit + 9,6us + 72*8*Tbit + 9,6 us

  Tempo3 = 134,4 us

- Escravo 4, Saída:

  Tempo4 = 72*8*Tbit + 9,6 us

  Tempo4 = 67,2 us

Assim, a temporização do escalonamento será de:

[tempo us] [escravo]

0000 – 1 ; 0 = escravo 1

0136 - 3 ; 0+ Tempo escravo 1 = 0+136 us = 136 us

0270 - CPU ; T anterior + Tempo escravo 3 = 136 us + 134,4 us = 270,4 us

1270 - 2 ; T anterior + Tempo CPU = 270 us + 1000 us = 1270 us

1362 - 4 ; T anterior + Tempo escravo 2 = 1270 us + 92,8us = 1362,8us

O início do próximo ciclo se daria em:

T anterior + Tempo escravo 4 = 1362+67,2= 1429,2 us.

Então, o tempo total do ciclo seria: 1429,2 us = 1,43 ms.

A ordenação da comunicação pode ser feita pela análise do software envolvido na comunicação da rede, mas através do software escalonador os escravos com entrada já são temporizados na comunicação antes dos escravos com saída.

4.6.2. Análise do Escalonamento nos Escravos

Na Figura 46 há um exemplo de como o escalonamento pode beneficiar o tempo de resposta de um sistema de controle.

Mesmo o tempo de ciclo sendo o mesmo, se as variáveis de entrada forem lidas antes das variáveis de saída, o tempo entre uma variação num dado de entrada em relação ao respectivo acionamento de uma saída será menor.

Quando menor for o tempo de processamento da CPU neste exemplo, maior será a eficiência do escalonamento bem como para sistemas que necessitem de tempo ciclo baixos como controle de movimento.

No caso do sistema proposto na Figura 46, há oito variáveis de entrada para o processamento de quatro blocos. Para estes blocos existem suas respectivas saídas, sendo uma saída por bloco.

Figura 46 - Comunicação Normal x Escalonada

Da forma como foi estruturada a comunicação sem escalonamento, o menor tempo entre a variação de uma entrada para sua respectiva saída seria de 22 intervalos de tempo.

Para a comunicação escalonada, este tempo seria de 16 intervalos de tempo.

No caso da comunicação normal, sem escalonamento, caso haja uma variação em uma entrada (representado por “X”) no instante em que a rede lê o escravo de entrada, esta variação somente estará presente na saída (representado por “T”) num próximo ciclo de varredura. Com escalonamento, a variação de entrada será executada no mesmo ciclo.

Nos dois exemplos, caso haja uma variação de entrada num instante após a rede ter lido o escravo de entrada, esta variação somente será lida no próximo ciclo. Mesmo nesta condição o escalonamento se mostra com melhor eficiência, pois o dado estará presente na saída seis instantes de tempo antes.

Na seqüência serão analisados os tempos de respostas de outras redes com processamento centralizado.

4.7. Análise de outras Redes

Para a análise de outros sistemas será necessário o uso de outros analisadores de protocolos. Existem alguns disponíveis no mercado.

Serão mostradas algumas informações de alguns Fielbuses bem como a análise de seus comportamentos.

4.7.1. Analisador para Rede Devicenet

O X-Analyser [52] é um analisador de protocolo para redes CAN.

A rede Devicenet [53], usa a camada física CAN e o protocolo de aplicação CIP, o mesmo usado na Ethernet/IP [9],

A Figura 47 mostra uma captura feita com o X-Analyser em uma rede Devicenet com dois escravos e com uma interface de configuração de rede [54] operando em 500 Kbps, a taxa máxima para esta rede.

Na Figura 47, a indicação "- - " indica que o endereço é o do mestre. Da mesma forma "01" é o escravo 1 e "06" escravo 6.

Como há somente dois escravos, o ciclo completo será:

"- - 01" - Mestre envia dados para escravo 01 (dados de saída)

"01 - -" – Escravo responde para Mestre (dados de entrada)

"- - 06" - Mestre envia dados para escravo 01 (dados de saída)

"06 - -" – Escravo responde para Mestre (dados de entrada)

O programa feito é:

- Bit 0 da saída do Escravo 1 recebe bit 0 da entrada do escravo 6

Pode-se notar que:

- O tempo de ciclo (tempo entre dois "- - 01") é de, pegando um ciclo como exemplo, 160.912863 – 160.901001 = 11,662 milissegundos.
- O tempo para a comunicação com cada escravo não é constante, num momento, por exemplo, é de 160.901233 – 160.901001 = 232 us em uma resposta do escravo à solicitação do mestre e em outro é de 160.901349 - 160.901233 = 116 us para um novo pedido do mestre.

A estrutura do CLP usado foi um cartão Devicenet numa CPU SLC5/05 da Rockwell, Allen-Bradley.

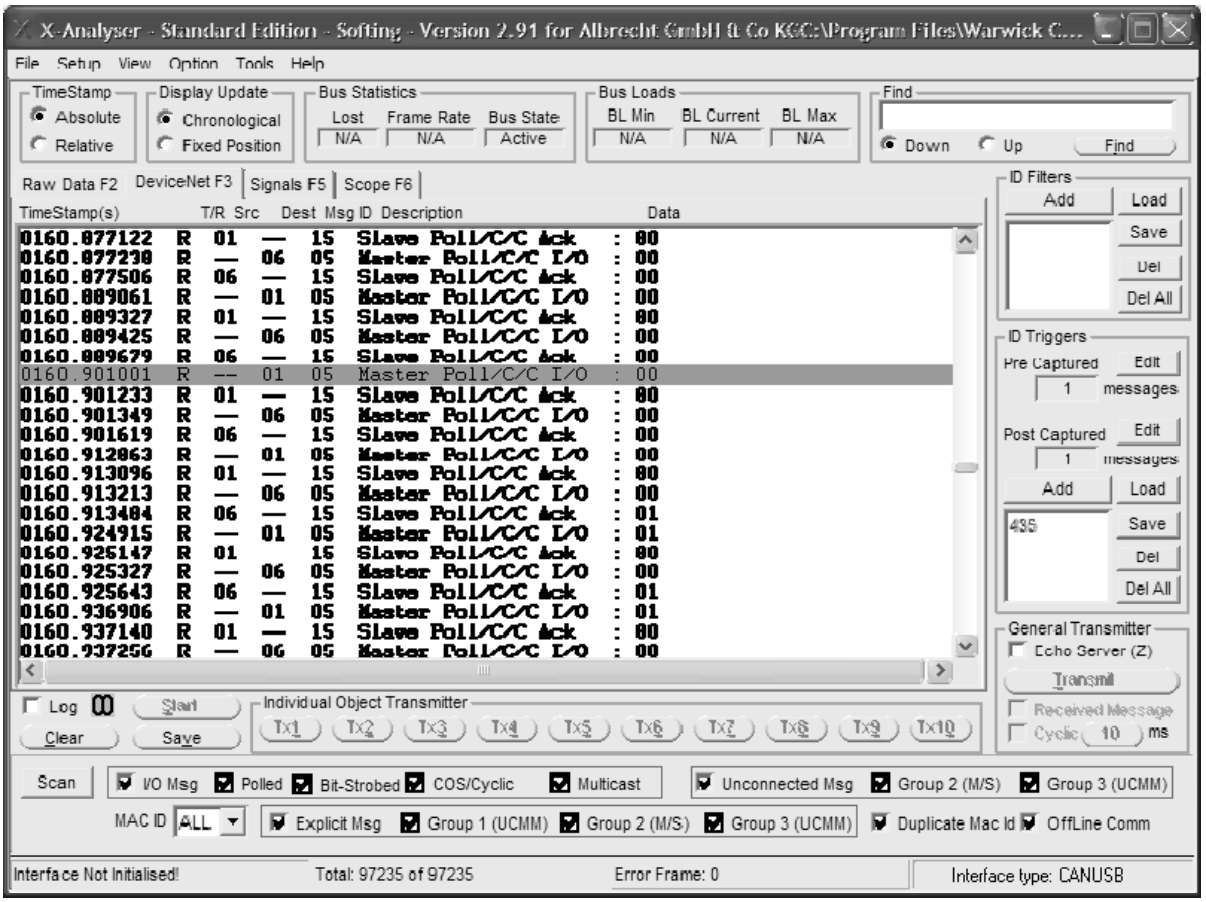

Figura 47 - Frame Devicenet Capturado

Da análise acima, pode-se notar que o atraso no sistema não é provocado pela varredura do cartão de rede Devicenet, mas sim o programa na CPU. Assim, mesmo se fosse usado um cartão Ethernet, neste caso Ethernet/IP que é um padrão da Rockwell (Allen-Bradley), o tempo de resposta do sistema não iria apresentar melhora. Mesmo sendo o Ethernet/IP ainda mais rápido, o processamento da CPU influenciaria no tempo de ciclo da rede.

4.7.2. Analisador para Rede Profibus

Para o Profibus DP, um analisador disponível no mercado é o Mobile PROFIBUS Analyzer [55] mostrado na Figura 48.

No exemplo a rede trabalha a uma taxa de 1,5Mbps.

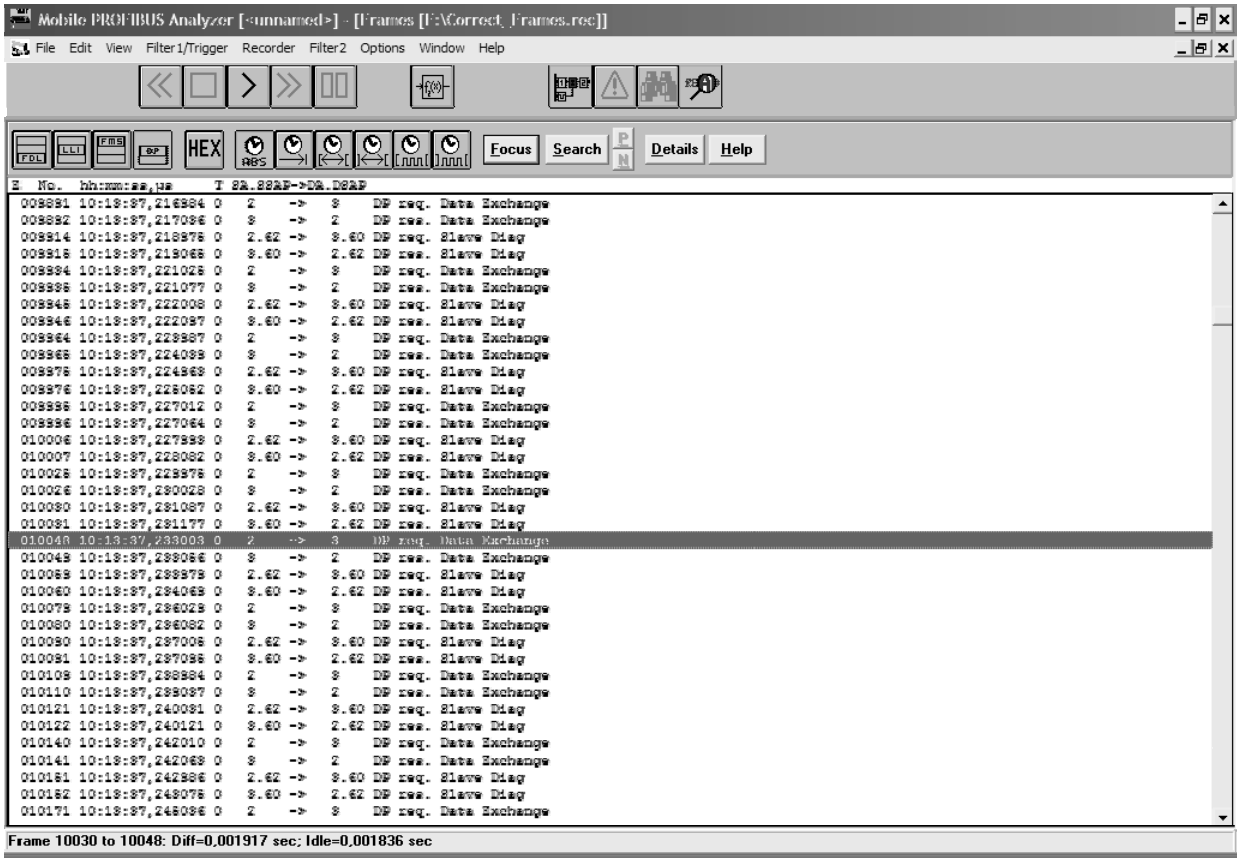

Figura 48 - Analisador Profibus DP

O tempo de ciclo para este exemplo, com somente um escravo na rede, foi de:

Tempo de Ciclo = 0,236029 – 0,233003 = 3,26 milissegundos.

Ou seja, o tempo entre dois "DP req. Data Exchange".

Da mesma forma que o exemplo de Devicenet, a rede se apresenta rápida, mas o tempo de ciclo é alto devido ao processamento da CPU.

4.7.3. Analisador para Rede Ethernet/IP

A ODVA [56] tem dois softwares que simulam um Cliente e um servidor da rede Ethernet/IP. As Figura 49 e Figura 50 mostram estes dois softwares.

Da análise com Ethereal, de acordo com a Figura 51, pode-se notar que a comunicação para troca de dados é feita via UDP.

Na Figura 50, esta comunicação está sendo usada para incrementar os indicadores coloridos (com valor instantâneo de 0038 em hexadecimal, Figura 50). A velocidade de atualização máxima encontrada foi de 2 Hz, ou seja, 500 milissegundos.

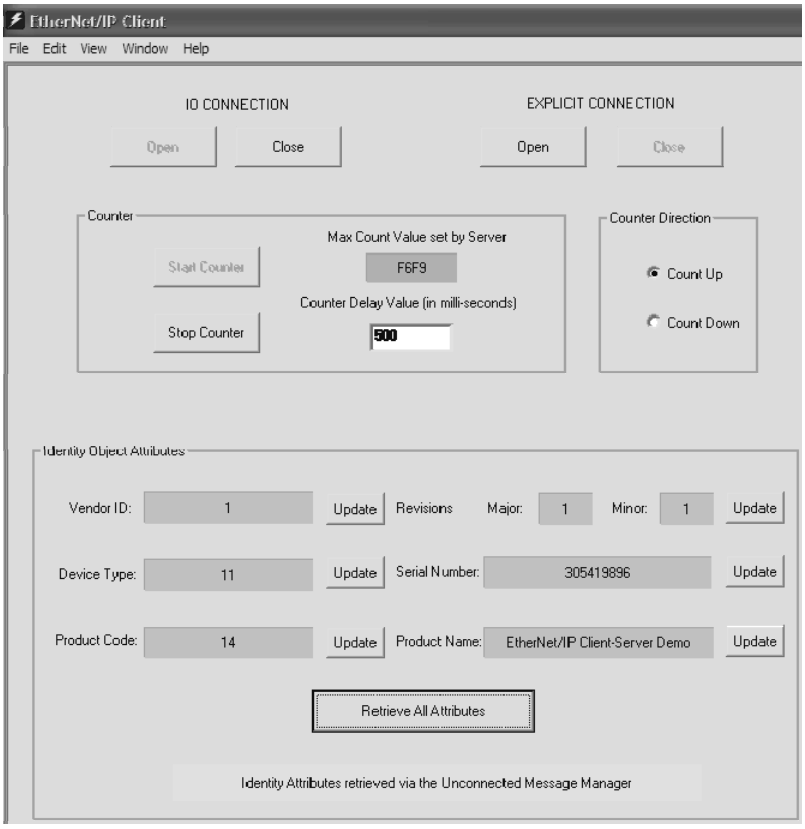

Figura 49 - Ethernet/IP Client

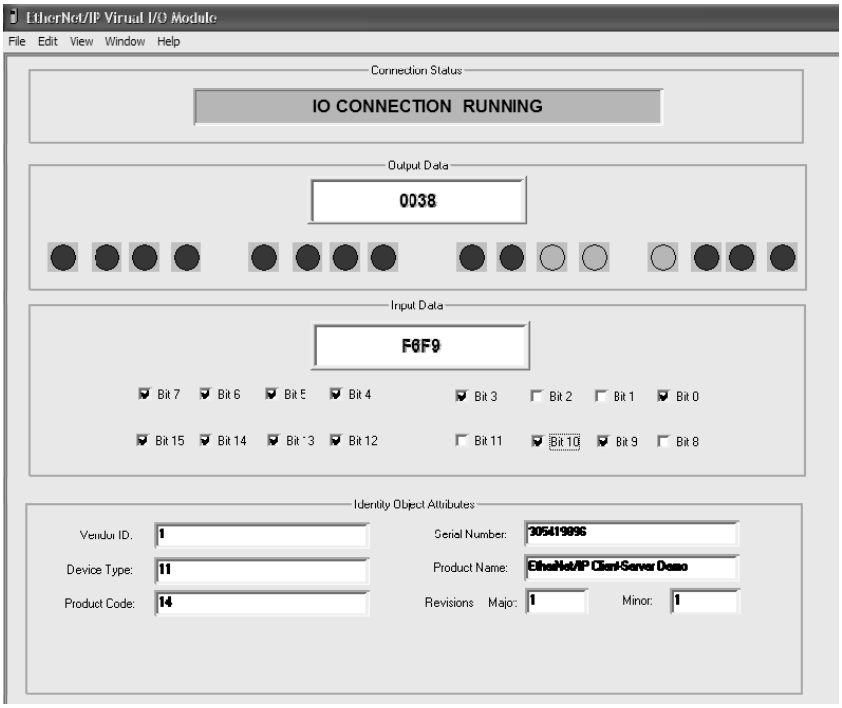

Figura 50 - Módulo I/O Ethernet/IP

Para o envio destes mesmos 10 bytes, foi necessário um frame de 70 bytes válidos na rede mostrado na Figura 51. Na verdade, o Ethereal não informa os dados de preâmbulo, assim, no total para estes 10 bytes foram enviados 78 bytes (70 + 8 de preâmbulo).

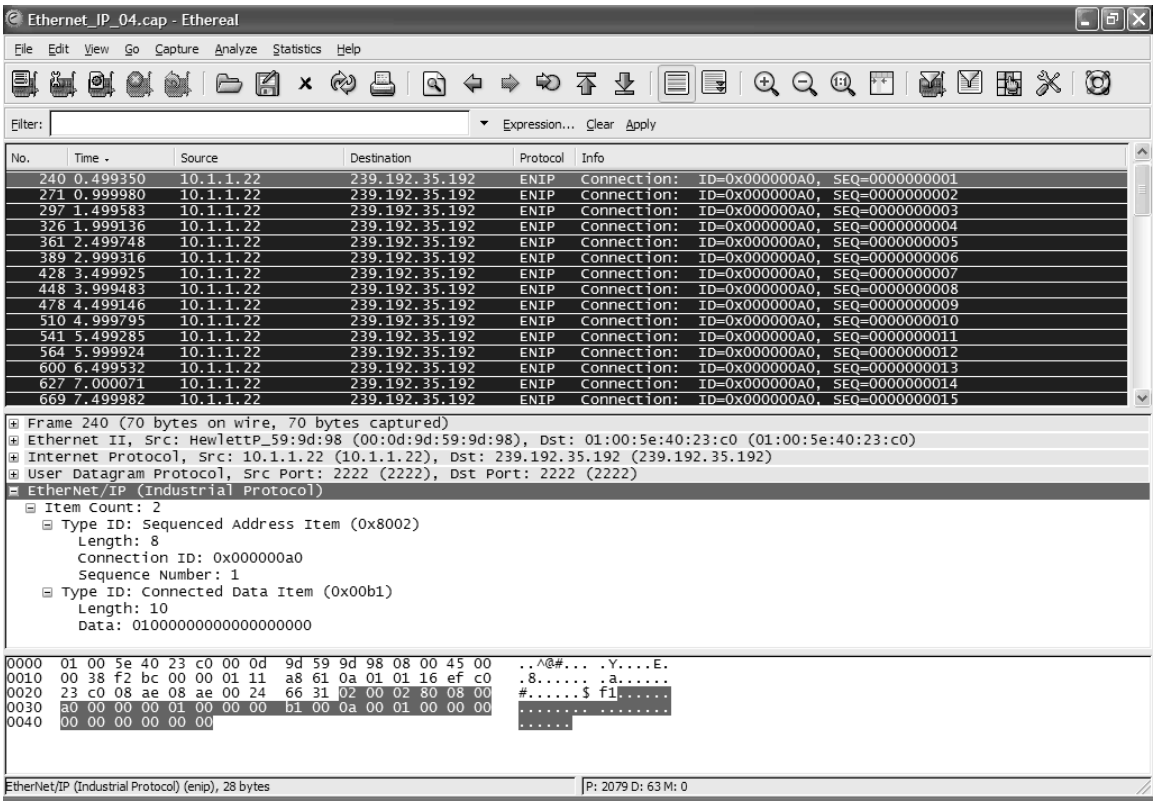

Figura 51 - Ethernet/IP no Ethereal

5. RESULTADOS

5.1. Os Resultados Previstos

De acordo com todos os cálculos e observações feitas, espera-se que o sistema simulado apresente uma eficiência maior no tempo de reação entre um dado de entrada e seu respectivo acionamento em um escravo de saída. Além disto, é esperado também um baixo tempo de ciclo. Isto será feito através da otimização no processamento da CPU que foi implementado no software.

Como forma de se obter um melhor desempenho, no exato momento em que todos os escravos de entradas são lidos, a CPU executará o programa de controle. E no momento em que a CPU executa toda a lógica de controle, todos os escravos de saída são atualizados. Desta forma o tempo entre o final da leitura das entradas e a atualização das saídas será somente o necessário para execução da lógica de controle.

5.2. Dificuldades encontradas

Uma das principais dificuldades encontradas foi executar o software do mestre e do escravo em uma plataforma que apresentasse uma comunicação rápida na rede Ethernet.

Num primeiro momento o software foi desenvolvido em Visual C++ utilizando a plataforma Windows. Porém, devido a limitações de tempo de resposta e exigências de tempo na grandeza de microssegundo para a aplicação dos softwares propostos neste trabalho, esta versão em Visual C++ não se apresentou eficiente visto que os tempos mínimos alcançados foram na casa de dezenas de milissegundos.

Como segunda opção o Java foi utilizado e apresentou bons resultados.

Porém a plataforma Windows não deu suporte a uma comunicação muito rápida na rede Ethernet. A transmissão de mensagem usando a placa de rede na interface Windows mostrou-se muito lenta.

Desta forma foi testada e aprovada a máquina virtual do Java aplicada ao ambiente Linux que apresentou os melhores resultados.

Para a análise do tráfego foi possível executar o Ethereal na versão original Linux.

Outro problema foi que o Ethereal não mostra algumas informações irrelevantes. Alguns dados de preenchimento usados apenas para se conseguir o tamanho mínimo na

transmissão (46 bytes no campo de dados do IEEE 802.3) não são mostrados. As informações sobre o CRC (checksum) também não foram encontradas no Software. Assim, para um frame de 72 Bytes a interface Ethereal disponibiliza somente 54 bytes.

Além disto, a bufferização dos dados a serem transmitidos gerou atrasos nos pacotes. Quando se faz várias transmissões em seqüência, a placa de rede exige um maior processamento e os dados que não são transmitidos instantaneamente vão para um buffer para posterior transmissão. Isto gera atraso caso o hardware da placa de rede receba vários pedidos de resposta e não consiga fazer a transmissão dos dados naquele momento, assim o valor do dado estará desatualizado. Para se resolver este problema a melhor solução foi à utilização de vários computadores de forma a se exigir um menor processamento de cada máquina.

5.3. Os Resultados Encontrados

Com os softwares sendo executados em uma rede real e também com troca de dados é possível analisar o tráfego na rede.

Várias simulações para diferentes condições foram feitas e serão detalhadas na seqüência.

5.3.1. Primeira Simulação

Na primeira simulação (Figura 52) o mestre está enviando pedidos de requisição para toda a rede, do escravo 1 ao escravo 15. Nesta simulação não há presença de escravos.

Na Figura 52 podem-se ver os frames capturados na rede e a temporização testes frames. A partir da marcação “REF” (que indica que o frame é a referência de tempo) há os instantes de tempo dos próximos dados transmitidos na rede.

O frame marcado é o último escravo de entrada, o escravo 15 (“Send data 15”). Para se comunicar com 15 escravos de entrada, o mestre gastou aproximadamente 1,2 milissegundos. E para se comunicar com todos os escravos, o tempo total foi de 2,52 milissegundos conforme mostrado na Figura 53.

O tempo entre uma comunicação de um escravo e outro foi de aproximadamente 72 microssegundos.

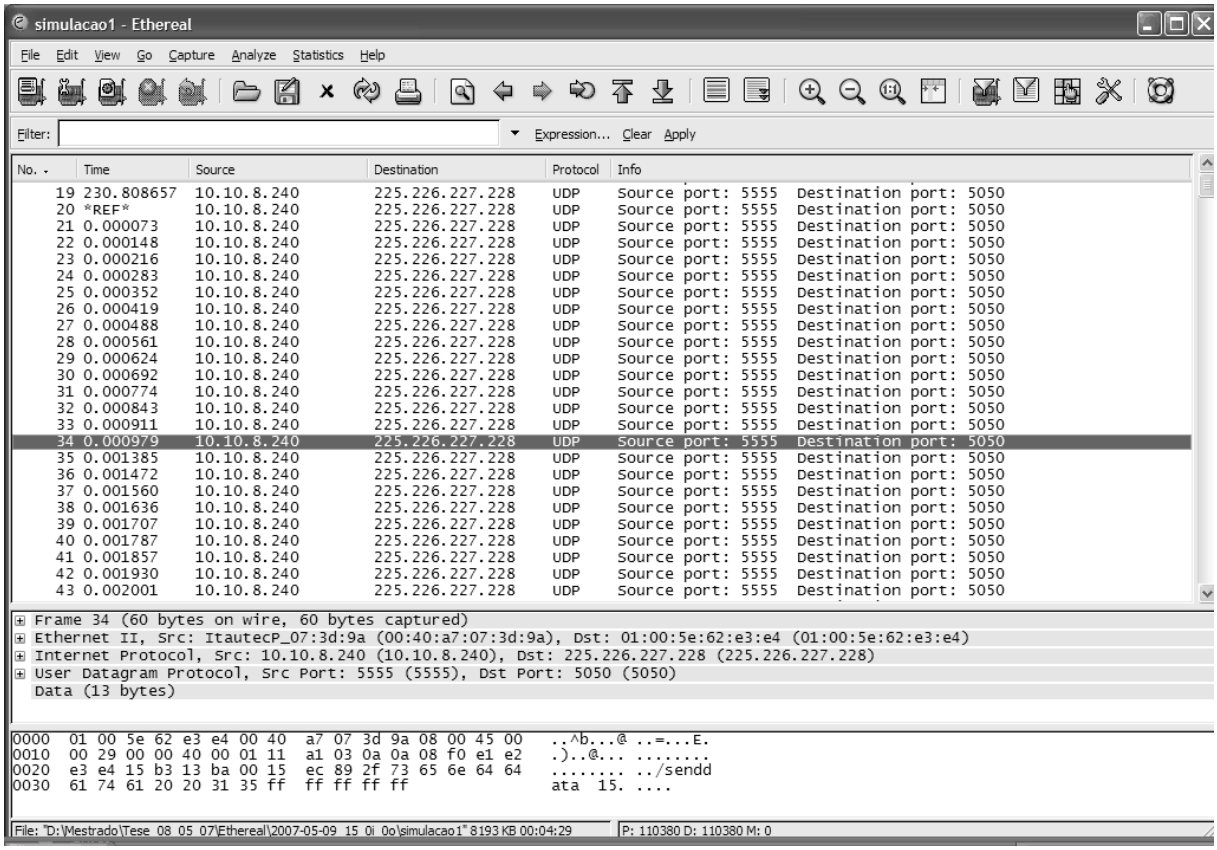

Figura 52 - Simulação 1 - Ciclo Parcial

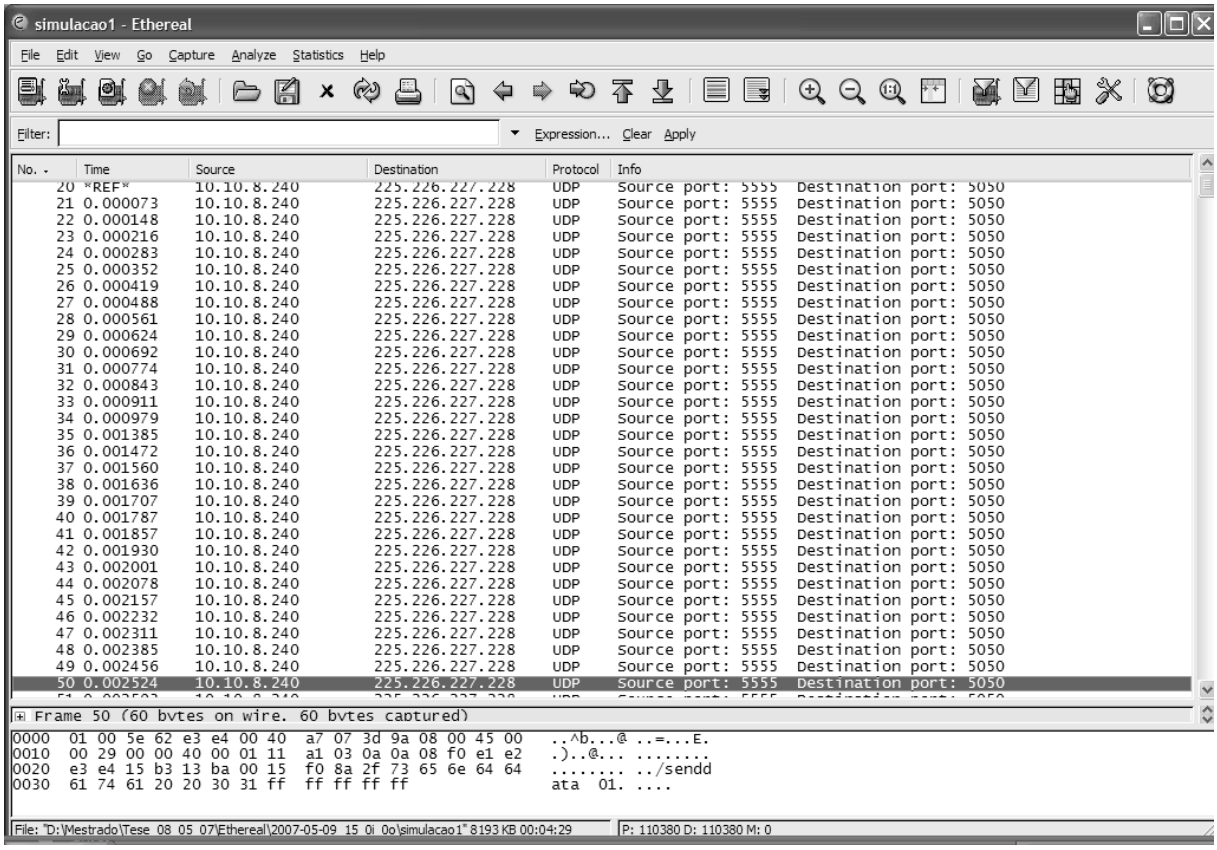

Figura 53 - Simulação 1 - Ciclo total

Nesta simulação a rede usada foi de 10 Mbps e assim o tempo entre a comunicação deveria ter sido de:

- 72 bytes * 8 bits * (1/10Mega) + 9,6 us = 67 us.

Ou seja, para uma taxa de 10 Mbps, o software do mestre encontra-se bem próximo do tempo real.

Por exemplo, para se comunicar com 15 escravos de entrada e 15 escravos de saída o tempo foi de 2,52 milissegundos. O valor teórico seria de: 67 us * 30 = 2,016 ms.

5.3.2. Segunda Simulação

Para uma segunda simulação tem-se um a rede com 5 escravos de entrada e 5 escravos de saída todos os softwares sendo executados no mesmo computador.

A taxa da rede é de 10 Mbps.

A CPU executará a seguinte lógica:

Escravo 6 = Escravo 1 + Escravo 2

Escravo 7 = Escravo 2 + Escravo 3

Escravo 8 = Escravo 3 + Escravo 4

Escravo 9 = Escravo 4 + Escravo 5

Escravo 10 = Escravo 5 + Escravo 1

As informações retiradas do relatório do Ethereal mostram:

No. Time	Source	Destination	Info
16 *REF*	10.10.8.240	225.226.227.228	/senddata 01.....
17 0.000199	10.10.8.240	225.226.227.228	/senddata 02.....
18 0.000311	10.10.8.240	225.226.227.228	/senddata 03.....
19 0.000393	10.10.8.241	225.226.227.228	/inputdata 1 01.
20 0.000437	10.10.8.235	225.226.227.228	/inputdata 8 02.
21 0.000571	10.10.8.239	225.226.227.228	/inputdata 39 03..
22 0.000611	10.10.8.240	225.226.227.228	/senddata 04.....
23 0.000768	10.10.8.240	225.226.227.228	/senddata 05.....
24 0.000863	10.10.8.236	225.226.227.228	/inputdata 18 04..
25 0.001011	10.10.8.242	225.226.227.228	/inputdata 3 05...
26 0.001324	10.10.8.240	225.226.227.228	/rcvdata 06 9.
27 0.001446	10.10.8.240	225.226.227.228	/rcvdata 07 47...
28 0.001579	10.10.8.240	225.226.227.228	/rcvdata 08 57..
29 0.001697	10.10.8.240	225.226.227.228	/rcvdata 09 21..
30 0.001822	10.10.8.240	225.226.227.228	/rcvdata 10 4..
31 0.001955	10.10.8.240	225.226.227.228	/senddata 01.....

Pode-se notar que o tempo de varredura total foi menos de 2 milissegundos para a rede com 10 escravos (1,955 milissegundos).

O endereço IP do mestre e dos escravos é o mesmo: 10.10.8.240, pois todos estão no mesmo computador.

Com este exemplo pode-se notar que o teste do sistema pode ser feito em um mesmo computador, mas para se aproximar das condições de simulação real de uma rede, esta condição não é viável. No caso de se utilizar um mesmo PC, a placa de rede será compartilhada e haverá dados fora da ordem normal de transmissão, como por exemplo, os frames 19, 20 e 21, dos dados acima, não deveriam estar nestas posições.

O tempo para execução da lógica:

Escravo 6 = Escravo 1 + Escravo 2

Foi de:

0.001324 - 0.000393 = 0,000931 segundos = 931 microssegundos.

No arquivo de escalonamento usado, o tempo de processamento da CPU foi colocado 1 microssegundo.

Esta simulação com apenas um computador foi feita para se determinar a influência do hardware no desempenho do software. Como pode ser notado, o uso de um único PC não trouxe bons resultados sendo então um cenário descartado para futuras simulações.

5.3.3. Terceira Simulação

Para uma melhor simulação podemos ter o mestre em um computador e os escravos em outros computadores.

Para esta simulação foi utilizado um computador para o mestre, dez para os dez escravos e mais um somente para analise de tráfego de rede. Com isto todos os processos estariam totalmente independentes.

A taxa da rede é de 100 Mbps.

Os endereços IPs foram:

Mestre: 10.10.8.240

Escravo 1: 10.10.8.241

Escravo 2: 10.10.8.235

Escravo 3: 10.10.8.239

Escravo 4: 10.10.8.236

Escravo 5: 10.10.8.242

Escravo 6: 10.10.8.243

Escravo 7: 10.10.8.245

Escravo 8: 10.10.8.244

Escravo 9: 10.10.8.248

Escravo 10: 10.10.8.247

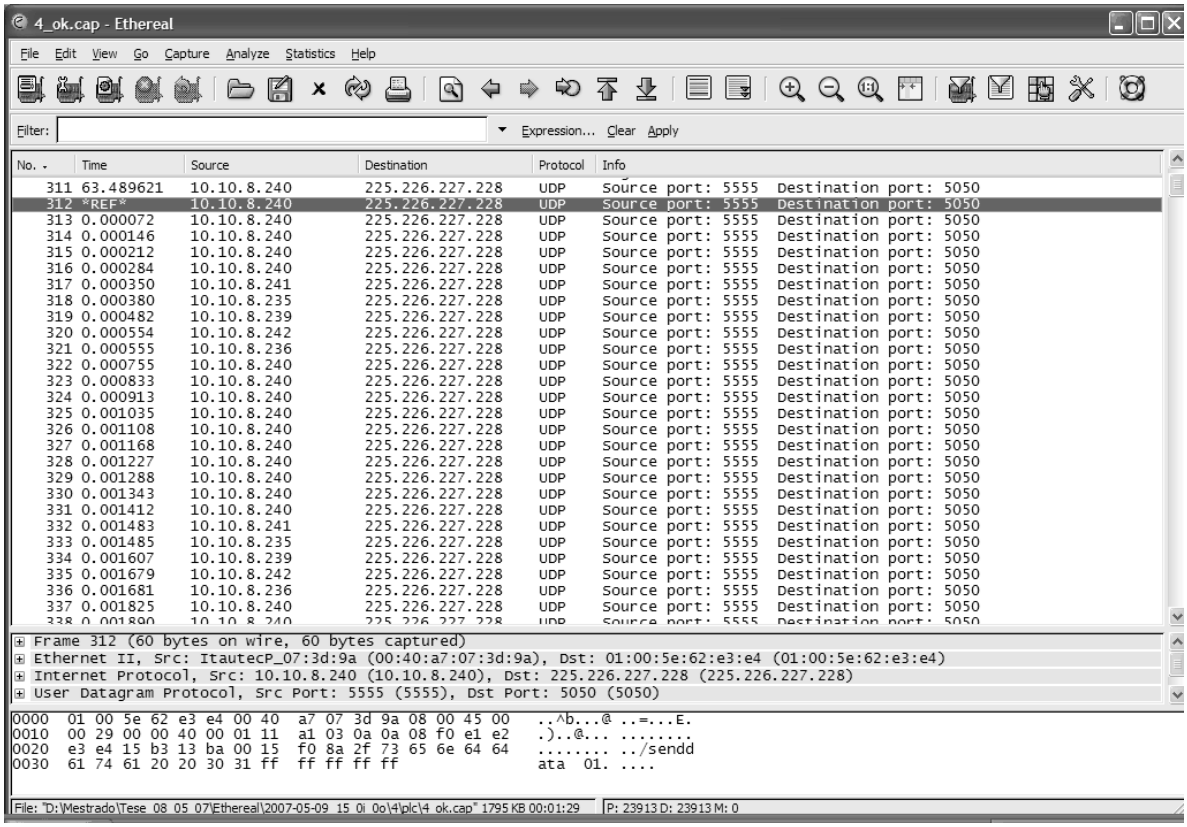

Figura 54 - Simulação 2 - Ciclo Total

A Figura 54 mostra a temporização dos frames capturados.

No software Ethereal pode-se ter informações mais detalhadas para cada frame, conforme os dados a seguir:

No. Time	Source	Destination	Info
312 *REF*	10.10.8.240	225.226.227.228	/senddata 01.....
313 0.000072	10.10.8.240	225.226.227.228	/senddata 02.....
314 0.000146	10.10.8.240	225.226.227.228	/senddata 03.....
315 0.000212	10.10.8.240	225.226.227.228	/senddata 04.....
316 0.000284	10.10.8.240	225.226.227.228	/senddata 05.....
317 0.000350	10.10.8.241	225.226.227.228	/inputdata 36 01.
318 0.000380	10.10.8.235	225.226.227.228	/inputdata 35 02.
319 0.000482	10.10.8.239	225.226.227.228	/inputdata 68 03..
320 0.000554	10.10.8.242	225.226.227.228	/inputdata 40 05..
321 0.000595	10.10.8.236	225.226.227.228	/inputdata 83 04..

// neste momento o programa será executado....

322 0.000755 10.10.8.240 225.226.227.228 /rcvdata 06 71.

323 0.000833 10.10.8.240 225.226.227.228 /rcvdata 07 103..

Analisando-se os dados acima vemos que:

- Coma a rede operando em 100 Mbps, o tempo entre frames, que antes era de aproximadamente 72 us, permaneceu bem próximo disto indicando que realmente se tem limitações de software e hardware. O tempo para 72 bytes deveria ter sido de 15,4 microssegundos.
- O tempo médio de cada ciclo foi de 4,140ms/4 = 1,035 ms.
- O tempo entre o acionamento do escravo de entrada até a sua resposta no escravo de saída foi de: Escravo 6 (322) = Escravo 1 (317) + Escravo 2 (318) = 755us- 350us = 405 microssegundos.
- O tempo alocado para o processamento da CPU foi de 1 microssegundo o que através do software do mestre é estabelecido para o menor possível, ou seja, quando se acaba a varredura dos escravos de entrada, o programa será executado e logo após o termino da execução os dados para os escravos de saída serão transmitidos.
- Antes de iniciar a execução do programa, o mestre aguarda todos os escravos responderem. Desta forma, os dados de saída serão corretamente atualizados.

5.3.4. Quarta Simulação

Na quarta simulação tem-se a comunicação sem escalonamento. Assim, o mestre comunica-se com todos os escravos de entrada de uma só vez, comunica com os escravos de saída e só então executa o programa:

No. Time	Source	Destination	Info
18 *REF*	10.10.8.240	225.226.227.228	/senddata 01.....
19 0.000070	10.10.8.240	225.226.227.228	/senddata 02.....
20 0.000148	10.10.8.240	225.226.227.228	/senddata 03.....
21 0.000214	10.10.8.240	225.226.227.228	/senddata 04.....
22 0.000280	10.10.8.240	225.226.227.228	/senddata 05.....
23 0.000340	10.10.8.241	225.226.227.228	/inputdata 36 01.
24 0.000370	10.10.8.235	225.226.227.228	/inputdata 35 02.
25 0.000472	10.10.8.239	225.226.227.228	/inputdata 68 03..
26 0.000548	10.10.8.242	225.226.227.228	/inputdata 40 05..
27 0.000608	10.10.8.236	225.226.227.228	/inputdata 83 04..

28 0.000722	10.10.8.240	225.226.227.228	/rcvdata 06 0.
29 0.000790	10.10.8.240	225.226.227.228	/rcvdata 07 0.
30 0.000860	10.10.8.240	225.226.227.228	/rcvdata 08 0.
31 0.000932	10.10.8.240	225.226.227.228	/rcvdata 09 0.
32 0.001028	10.10.8.240	225.226.227.228	/rcvdata 10 0.

// neste momento o programa será executado....

33 0.001194	10.10.8.240	225.226.227.228	/senddata 01.....
34 0.001264	10.10.8.240	225.226.227.228	/senddata 02.....
35 0.001338	10.10.8.240	225.226.227.228	/senddata 03.....
36 0.001404	10.10.8.240	225.226.227.228	/senddata 04.....
37 0.001472	10.10.8.240	225.226.227.228	/senddata 05.....
38 0.001490	10.10.8.241	225.226.227.228	/inputdata 37 01..
39 0.001504	10.10.8.235	225.226.227.228	/inputdata 36 02..
40 0.001577	10.10.8.239	225.226.227.228	/inputdata 69 03..
41 0.001602	10.10.8.242	225.226.227.228	/inputdata 41 05..
42 0.001708	10.10.8.236	225.226.227.228	/inputdata 84 04..
43 0.001798	10.10.8.240	225.226.227.228	/rcvdata 06 71.

Comparando com a simulação anterior com escalonamento:

- Sem o escalonamento, o tempo entre o acionamento do escravo de entrada até a sua resposta no escravo de saída foi de: Escravo 6 (43) = Escravo 1 (23) + Escravo 2 (24) = 1798 us- 340 us = 1,458 ms.
- Com escalonamento, este mesmo tempo foi de 405 microssegundos.
- A ocupação no barramento nas últimas simulações aconteceu de forma controlada, pois a situação foi a mais favorável possível. Sendo o tempo entre frames grande, há possibilidade de tráfego para outros sistemas.

5.3.5. Quinta Simulação

Considerando o tempo de processamento de 1 milissegundo da CPU para duas situações, escalonado e não escalonado, e também uma nova lógica de controle:

- Escavo 9 = Escravo 2 + Escravo 6;
- Escravo de entrada: Escravo 2, 3, 4, 5, 6 e 10;
- Escravo de Saída: Escravo 1, 7,8 e 9.

Sem escalonamento:

No. Time	Source	Destination	Info
12 *REF*	10.10.8.240	225.226.227.228	/rcvdata 01 0.
13 0.000072	10.10.8.240	225.226.227.228	/senddata 02.....

14 0.000140	10.10.8.240	225.226.227.228	/senddata 03.....
15 0.000210	10.10.8.240	225.226.227.228	/senddata 04.....
16 0.000282	10.10.8.240	225.226.227.228	/senddata 05.....
17 0.000340	10.10.8.240	225.226.227.228	/senddata 06.....
18 0.000424	10.10.8.240	225.226.227.228	/rcvdata 07 0.
19 0.000492	10.10.8.240	225.226.227.228	/rcvdata 08 0.
20 0.000558	10.10.8.240	225.226.227.228	/rcvdata 09 0.
21 0.000622	10.10.8.240	225.226.227.228	/senddata 10.....
22 0.000676	10.10.8.241	225.226.227.228	/inputdata 36 03..
23 0.000700	10.10.8.235	225.226.227.228	/inputdata 07 02.
24 0.000758	10.10.8.239	225.226.227.228	/inputdata 68 04..
25 0.000792	10.10.8.242	225.226.227.228	/inputdata 40 05..
26 0.000865	10.10.8.236	225.226.227.228	/inputdata 03 06.
27 0.000954	10.10.8.236	225.226.227.228	/inputdata 83 10..
// neste momento o programa será executado....			
28 0.002008	10.10.8.240	225.226.227.228	/rcvdata 01 10.
29 0.002082	10.10.8.240	225.226.227.228	/senddata 02.....
30 0.002142	10.10.8.240	225.226.227.228	/senddata 03.....
31 0.002222	10.10.8.240	225.226.227.228	/senddata 04.....
32 0.002286	10.10.8.240	225.226.227.228	/senddata 05.....
33 0.002360	10.10.8.240	225.226.227.228	/senddata 06.....
34 0.002432	10.10.8.240	225.226.227.228	/rcvdata 07 0.
35 0,002500	10.10.8.240	225.226.227.228	/rcvdata 08 0.
36 0,002568	10.10.8.240	225.226.227.228	/rcvdata 09 10.

Dos dados acima, tem-se que:

- Sem o escalonamento, o tempo entre o acionamento do escravo de entrada até a sua resposta no escravo de saída foi de: Escravo 9 (36) = Escravo 2 (23) + Escravo 6 (26) = 2568 us- 700 us = 1,868 ms.

A mesma condição, mas com escalonamento:

No. Time	Source	Destination	Info
10 *REF*	10.10.8.240	225.226.227.228	/senddata 02....
11 0.000074	10.10.8.240	225.226.227.228	/senddata 03.....
12 0.000140	10.10.8.240	225.226.227.228	/senddata 04.....
13 0.000214	10.10.8.240	225.226.227.228	/senddata 05.....
14 0.000278	10.10.8.240	225.226.227.228	/senddata 06.....
15 0.000350	10.10.8.240	225.226.227.228	/senddata 10....
16 0.000370	10.10.8.241	225.226.227.228	/inputdata 36 02.
17 0.000392	10.10.8.235	225.226.227.228	/inputdata 07 03..
18 0.000442	10.10.8.239	225.226.227.228	/inputdata 06 05..

19 0.000500 10.10.8.242 225.226.227.228 /inputdata 01 04..

20 0.000554 10.10.8.236 225.226.227.228 /inputdata 03 06.

21 0.000600 10.10.8.236 225.226.227.228 /inputdata 83 10

// tempo de aproximadamente 1 milissegundo

22 0.001704 10.10.8.240 225.226.227.228 /rcvdata 01 0.

23 0.001774 10.10.8.240 225.226.227.228 /rcvdata 07 0.

24 0.001840 10.10.8.240 225.226.227.228 /rcvdata 08 0.

25 0.001924 10.10.8.241 225.226.227.228 /rcvdata 09 39.

Dos dados acima, tem-se que:

- Com escalonamento, o tempo entre o acionamento do escravo de entrada até a sua resposta no escravo de saída foi de: Escravo 9 (25) = Escravo 2 (16) + Escravo 6 (20) = 1924 us- 370 us = 1,554 ms.

5.3.6. Conclusões das Simulações

Conclusões das simulações feitas:

- Da primeira simulação, temos que o software apresenta um comportamento bem próximo da comunicação real para uma rede em 10 Mbps;
- Da segunda simulação, temos que uma distribuição do processamento pode apresentar melhores resultados que o uso de um único computador;
- Na terceira simulação, com a comunicação escalonada e o tempo de processamento da CPU otimizado obteve-se um bom resultado;
- Na quarta simulação, diferentemente da terceira, o resultado não foi tão bom, o que comprova que o escalonamento realmente melhora a eficiência do sistema.
- Da quinta simulação, vemos que o escalonamento apresentou bons resultados. Porém quanto maior for o tempo de execução do programa de controle da CPU menor será o seu rendimento. Por outro lado, quanto menor o tempo de ciclo necessário para uma aplicação, menor terá que ser o tempo de execução da CPU do CLP e assim maior o rendimento usando-se o escalonamento.

6. CONTRIBUIÇÕES E CONCLUSÕES

Com a evolução natural da microeletrônica, os CLPs tenderão a ser cada vez mais rápidos e eficientes atendendo cada vez mais às aplicações com baixo ciclo tempo. E assim quanto menor for o tempo de processamento da CPU do CLP, melhor será a eficiência do escalonamento.

A conclusão importante deste trabalho foi mostrar que a CPU de um CLP compromete a eficiência do sistema mesmo com uma rede de comunicação rápida. Para aplicações de baixo tempo de ciclo é necessária além de uma rede rápida uma CPU rápida.

Com o uso da rede Ethernet Industrial para tráfego de poucos bytes, o desempenho da rede acaba por não ser alto, pois a rede Ethernet foi projetada para o tráfego de muitos bytes.

Se antes os Fieldbuses estavam preocupados com sua própria evolução, agora deverão também estar preparados para a Ethernet que está cada vez mais rápida e com grande capacidade de absorção destes mesmos fieldbuses.

Conforme mostrado nos resultados, um controle de processamento na varredura da rede e na temporização do processamento da CPU do CLP tem grandes resultados chegando-se a um sistema mais ativo, menos ocioso e com uma melhor ocupação do barramento. Isto resulta num melhor tempo de resposta do sistema de controle favorecendo aplicações com baixos tempos de ciclo.

Os problemas encontrados no desenvolvimento deste trabalho, como o tempo de processamento, mostram a necessidade cada vez maior de se ter sistemas com hardware e softwares mais eficientes. Os hardwares deverão ser cada vez mais eficientes e não só na comunicação com a rede, mas também nos processamentos dos dados de entrada e de saída de um dispositivo no chão-de-fábrica.

Se há um cartão de rede Devicenet, por exemplo, fazendo o controle de um processo e se este cartão for trocado por cartão Ethernet/IP no CLP, a rede será bem mais rápida que a anterior Devicenet. Porém, não se terá necessariamente uma melhora no tempo de resposta do sistema de controle, pois na maioria das vezes, o que compromete o tempo no sistema é a CPU do CLP e não a rede envolvida na comunicação. Por outro lado, se for trocado o CLP por outro com uma CPU mais rápida pode-se ter uma melhora no sistema. E com uma CPU mais rápida a rede Ethernet poderá inclusive ser escalonada o que apresentaria uma melhoria ainda mais significativa.

O escalonamento proposto neste trabalho, apesar de não ser muito efetivo para sistemas convencionais onde os processos não necessitam de velocidade, pode ser utilizado com bom desempenho em aplicações de controle de movimento e sincronização de inversores que se caracterizam pela necessidade de tempo de resposta menores. Da mesma forma, com a evolução das indústrias e da tecnologia no chão-de-fábrica, o escalonamento poderá ser empregado em demais aplicações que necessitem também de um baixo tempo de ciclo.

7. TRABALHOS FUTUROS

A partir deste trabalho, pode-se evoluir o conceito de se ter uma rede mais eficiente que apresente uma ocupação menor do barramento com maior eficiência de processamento para sistemas que necessitem de atualizações rápidas de variáveis.

A Ethernet operando em 1 Gbps está cada vez mais próxima, inclusive do chão-de-fábrica. Mas para se ter um bom desempenho é necessário que todos os dispositivos envolvidos no processamento destes dados estejam no mesmo grau de evolução. A rede Ethernet tem uma forte evolução de software e hardware e agora com esta aproximação do chão-de-fábrica é necessário também que todos os barramentos estejam preparados para trabalhar nesta nova tecnologia.

Com o switch priorizando o tráfego e com os mestres da rede ocupando o barramento somente o necessário para executar seus programas de controle, outros tráfegos poderão acontecer de forma que nenhum dos sistemas envolvidos sofra qualquer prejuízo, principalmente em sistemas com baixo tempo de resposta.

Com base no escalonamento proposto, podem ser feitos outros trabalhos também em redes não só com o sistema cíclico de varredura, mas também sistemas que tenham outros meios de comunicação como Change of State (COS) do Devicenet, por exemplo.

Com o avanço da microeletrônica, há atualmente microcontroladores com hardwares capazes de operar diretamente na Ethernet e também executar a máquina virtual do Java. Assim, a partir dos softwares feitos neste trabalho seria possível ter um hardware que executasse (com algumas alterações), os softwares do mestre e do escravo.

Pode-se também adequar o software utilizado nas simulações para a camada de aplicações do Ethernet/IP alterando somente a camada mais alta, pois as camadas mais baixas usadas são as mesmas.

8. REFERÊNCIAS BIBLIOGRÁFICAS

[1] LOPEZ, Ricardo Aldabó. **Sistemas de Redes para controle e Automação.** Editora Express. 2000. pp 89-92.

[2] http://www.isa.org/ **International Society for Measurement and Control.** Acesso em Jan. 2007.

[3] http://www.iec.org/ **International Engineering Consortium.** Acesso em Fev. 2007.

[4] http://www.profibus.com/ **Profibus: The Industrial Communications Community Delivering Greater Enterprise Advantage**. Acesso em Jan. 2007.

[5] FRENCH ASSOCIATION FOR STANDARDIZATION. FIP. Bus for Exchange of Information Between Transmitters, Actuators and Programmable Controllers, NF C46601-C46607, 1989--1992.

[6] FELSER, Max, SAUTER, Thilo. **The Fieldbus War: History or Short Break Between Battles?** 4th IEEE International Workshop on Factory Communication System, Vasteras, Sweded. Ago. 2002, pp 73- 80.

[7] IEC, **Digital data communications for measurement and control,** IEC 61784-1, First edition, Mai. 2003.

[8] POSCHMANN, A.; NEUMANN, P. **Institut fur Automation und Kommunikation Magdeburg Architecture and Model of Profinet.** Germany. IEEE 2004.

[9] BROOKS, Paul. **Ethernet/IP – Industrial Protocol–Logix/NetLinx Technology Adoption**, Rockwell Automation's European Marketing Manager, Belgium, IEEE 2001.

[10] 802.3 IEEE Standard for Information technology Telecommunications and information exchange between systems. **Local and metropolitan area networks — Specific requirements Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications**. IEEE 2002.

[11] DECOTIGNIE, Jean Dominique. **A perspective on Ethernet-TCP/IP as a fieldbus.** IFAC International Conference on Fieldbus Systems and their Applications, 2001. pp 138-142.

[12] CRWKETT, Neil. **Connecting the Factory Floor.** Business Development Director, Manufacturing Sector, Cisco Systems EMEA**.** IEE Manufacturing Engeneer, Jun. 2003, pp 41.

[13] Hubert Zimmermann, **The ISO Model of Architecture for Open Systems Interconnection.** IEEE Transactions on Communications, vol. 28, no. 4, Abr. 1980, pp. 425 - 432.

[14] TANENBAUM, Andrew S. **Redes de Computadores**. Editora Campus, tradução da quarta edição, 2003, pp 39-42, 320-326, 471-474, 595-623.

[15] CERF, V; KAHN. **A Protocol for Packet Network Interconnection.** IEEE Tran. on Commum., Vol. COM-22, 1974, pp 637-648,.

[16] ACM SIGCOMM Computer Communication, **The Design Philosophy of the DARPA Internet Protocols.** California. United States. 1988, pp 106-114.

[17] INDUSTRIAL ETHERNET BOOK, edição 10. **Connectors for harsh conditions**. Jun. 2002, pp 7.

[18] INDUSTRIAL ETHERNET BOOK, edição 12. **Industrial 100Mbps connectors: signed, sealed and delivered?** Nov. 2002, pp 24.

[19] Postel, J., **Internet Protocol**, RFC 760, USC/Information Sciences Institute, January 1980.

[20] Postel, J., **Transmission Control Protocol**, RFC 761, USC/Information Sciences Institute, January 1980.

[21] Postel, J., **User Datagram Protocol**, RFC 768, USC/Information Sciences Institute, January 1980.

[22] Network Working Group, **Stream Control Transmission Protocol (SCTP)**, RFC2960, R. Stewart, Q. Xie, Motorola, K. Morneault, C. Sharp, Cisco, H. Schwarzbauer, Siemens, T. Taylor, Nortel Networks, I. Rytina, Ericsson, M. Kalla, Telcordia, L. Zhang, UCLA, V. Paxson, ACIRI, Out. 2000.

[23] Network Working Group, **Datagram Congestion Control Protocol (DCCP)**, RFC 4340, E. Kohler, UCLA, M. Handley, UCL, S. Floyd, ICIR, Mar. 2006.

[24] ETHERNET/IP. **Specification Release 1.0**. Disponível no site: www.odva.org. Acesso em Jun. 2001.

[25] www.ab.com/en/epub/catalogs/12762/2181376/214372/1810894/3404056/tab7.html **Allen Bradley**. Acesso em Ago. 2005.

[26] IEC/PAS 62411 Ed. 1.0:2005. **Real-time Ethernet PROFINET IO.** Jun. 2005

[27] www.sitrain.com/modules/profinet_en/times/index_1024.htm **Curso Siemens de Profinet**, Acesso em Abr. 2007.

[28] POPP, Manfred; WEBBWE, Karl. **The Rapid Way to Profinet**, Nov. 2004, pp 10-50.

[29] FELSER, Max. **Real-Time Ethernet – Industry Prospective**. Proceedings of the IEEE, edição 6. Jun. 2005. pp 1118 – 1129.

[30] www.automation.siemens.com/profinet/html_76/produkte/ertec_400.htm **Profinet, Siemens Automação**. Acesso em Dez. 2006.

[31] SMAR, **HSE - High Speed Ethernet**, Set. 2003.

[32] SILVA NETO, Eugenio F. da; BERRIE, Peter G. **High Speed Ethernet - Promoting openness in hybrid control**. Endress+Hauser Process Solutions AG, atp international. Abr. 2006, pp 39-45.

[33] Larsson, L., **Fourteen Industrial Ethernet Solutions Under the Spotlight**, The Industrial Ethernet Book, Edição 37, Set. 2005. ,
Disponível em: http://ethernet.industrial-networking.com/articles/articledisplay.asp?id=854

[34] QIZHI ZHANG,YUNZE CAI; DANYING GU; WEIDONG ZHANG; **Determine the Maximum Closed-Loop Control Delay in Switched Industrial Ethernet Using Network Calculus**. Proceedings of the 2006 American Control Conference Minneapolis. Minnesota, USA. Jun. 2006. pp 14-16.

[35] DECOTIGNIE, Jean Dominique; FELLOW; **Ethernet Based Real Time and Industrial Communications**; Proceedings Of the IEEE, Vol.93, No. 6, Jun. 2005, pp 1102- 1117.

[36] KROMMENACKER, Nicolas; RONDEAU, Eric. DIVOUX, Thierry. **Genetic Algorithms for Industrial Ethernet Network Design**. 4 th IEEE International Workshop on Factory Communication System. Vasteras. Sweden. Ago. 2002, pp 149- 156.

[37] DECOTIGNIE, Jean Dominique; KOUTHON, T. **Improving Time Performances of Distributed PLC Applications**. Conferência IEEE EFTA 1996, Vol. 2. pp 656-662.

[38] SEUNGKWEON JEONG; YOUNG SHIN KIM; WOOK HYUN KWON. **Scheduling Algorithm for Programmable Logic Controllers with Remote I/Os**. Fourth International Workshop on Real-Time Computing Systems and Applications (RTCSA), 1997, pp 87.

[39] FENG-LI LIAN; JAMES MOYNE; DAWN TILBURY; **Network Design Consideration for Distributed Control Systems**. IEEE Transaction on Control System Technology. Vol. 10. No. 2. Mar. 2002, pp 297-307.

[40] LEI FU; GUANZHONG DAÍ. **Delay characteristics and synchronization architecture of networked control system**. 1st International Symposium on Systems and Control in Aerospace and Astronautics. ISSCAA 2006. Jan. 2006, pp 4.

[41] Vyatkin, V. **Execution Semantic of Function Blocks based on the Model of Net Condition/Event Systems**. IEEE International Conference on Industrial Informatics. Aug. 2006, pp 874-879.

[42] LEE, Kyung Chang ; LEE, Suk; **Performance Evaluation of Switched Ethernet for Networked Control System** – IECON 02- 28th Annual Conference of the Industrial Electronics Society, IEEE 2002, Volume 4, pp 3170 – 3175.

[43] CUONG, Dao Manh; KIM, Myung Kyun; LEE, Hee Chan. **Supporting Hard Real-time Communication of Periodic Messages over Switched Ethernet,** The 1st International Forum on Strategic Technology, Out. 2006, pp 419-422.

[44] LEE, K.C.; LEE, S.; LEE, M.H. **Worst Case Communication Delay of Real-Time Industrial Switched Ethernet With Multiple Levels.** Transactions on Industrial Electronics, IEEE. Out. 2006; Vol. 53, Edição 5. pp 1669-1676

[45] VIEGAS, R.; VALENTIM, R.A.M.; TEXEIRA, D.G.; GUEDES, L.A. **Analysis of Protocols to Ethernet Automation Networks.** International Joint Conference SICE-ICASE. Out. 2006, pp 4981-4985.

[46] MOLDOVANSKY, Anatoly; **Utilization of Modern Switching Technology in Ethernet/IP Networks.** 1st Workshop On Real-Time Lans In The Internet Age. Technical University of Vienna. Austria. Jun. 2002.

[47] MORAES, Cícero Couto; CASTRUCCI, Plínio de Lauro. **Engenharia de Automação Industrial**; Editora LTC. 2001, pp 35-42, 109-133, 213-224.

[48] HELD, Gilbert**. Internetworking LANs and WANs.** Segunda Edição. Editora Wiley. 2003, pp 320-350.

[49] ILMENAU UNIVERSITY OF TECHNOLOGY. **Visual object oriented Petri Net based Engineering Tool.** Version 2.0a Germany. Copyright: Rainer Drath. Homepage: www.systemtechnik.tu-ilmenau.de/~drath/visual_e.htm. Acesso em Mai. 2003.

[50] Markov, A.A. **Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga**. Izvestiya Fiziko-matematicheskogo obschestva pri Kazanskom universitete, pp 135-156, 1906.

[51] ETHEREAL, **The world's most popular network protocol analyzer.** Homepage: www.ethereal.com. Acesso em Fev. 2007.

[52] Warwick Control Technologies Ltd. **X-ANALYSER.** U.K., version 2.91; www.x-analyser.com ou www.warwickcontrol.com. Acesso em Set. 2006.

[53] CIP implementation. **DEVICENET - device-level network for industrial automation**. DeviceNet é marca registrada da ODVA: Open Device Vendors Association: www.odva.org, Acesso em Jan. 2006.

[54] ROCKWELL AUTOMATION, **DeviceNet RS232 Interface Module**; Allen-Bradley www.rockwellautomation.com/index.html. Acesso em Jul. 2006.

[55] PROFI-MON-MOBIL, **Mobile PROFIBUS Analyzer**, Version 1.30.0.00; Softing AG, www.softing.com/home/en/industrial-automation/products/profibus-dp/diagnostic-tools/protocol-analyzer-mobile.php. Acesso em Set. 2006.

[56] ODVA: Open Device Vendors Association. **Example Code Freeware**, disponível em www.odva.org/Portals/0/Library/Publications_Numbered/ENetIP_EC_V7.1.zip
Acesso em Dez. 2006