UNIVERSIDADE FEDERAL DE VIÇOSA
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

GIOVANNI COSTA HENRIQUES

ANÁLISE DO CICLO DE VARREDURA DO CLIC-02 DA WEG

VIÇOSA
2018

GIOVANNI COSTA HENRIQUES

ANÁLISE DO CICLO DE VARREDURA DO CLIC-02 DA WEG

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.
Orientador: Prof. Dr. Alexandre Brandão

VIÇOSA
2018

GIOVANNI COSTA HENRIQUES

ANÁLISE DO CICLO DE VARREDURA DO CLIC-02 DA WEG

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Aprovada em 09 de julho de 2018.

COMISSÃO EXAMINADORA

Prof. Dr. Alexandre Santos Brandão - Orientador
Universidade Federal de Viçosa

M.Sc. Gerson Ovídio Luz Pedruzi - Membro
Universidade Federal de Viçosa

M. Sc. André Luis Carvalho Mendes - Membro
Universidade Federal de Viçosa

“Missão dada, parceiro, é missão cumprida.”

(Capitão Nascimento)

Dedico a todos que participaram da minha árdua caminhada durante a graduação.

Agradecimentos

Enfim é chegado o momento, hora de mudar de fase. E não foi nada fácil chegar até aqui, mas teria sido muito mais difícil se não fossem as pessoas maravilhosas que sempre tive ao meu lado. A começar pelos meus queridos pais, Roberto e Sônia, pelo apoio incondicional. Sem vocês seria impossível, e dedico essa vitória aos dois. Agradeço também aos meus irmãos, Guilherme e Gustavo, pela amizade e apoio diário, amenizando a saudade de casa. Ao cELTics, que levarei pra sempre comigo. À família Elétrica, todos os amigos e colegas que fiz nessa Universidade maravilhosa e que fazem parte dos momentos difíceis, dos dias e noites de estudo, e das comemorações. Aos professores da Universidade e da vida, que tanto me ensinaram nesses 6 anos de Viçosa. Ao meu professor orientador Alexandre Brandão, que me auxiliou com este trabalho e me fez apaixonar por automação. E à Luíza, que chegou para me dar forças na reta final. Porque missão dada, parceiro, é missão cumprida.

Resumo

A evolução dos processos industriais se deu de forma exponencial com a chegada dos computadores dedicados a automatizar as plantas industriais, estes denominados por Controlador Lógico Programável (CLP). Os CLP's permitem a execução de várias atividades de forma automática e contínua, em diversas áreas diferentes, inclusive aplicações que exigem uma leitura da planta (processo), e uma tomada de atitude bem veloz, na casa dos milissegundos. Por via de regra, esses controladores operam em tempo real, ou seja, pré-definido para execução de suas instruções, variando de acordo com o software programado. Neste contexto, o objetivo deste trabalho é verificar o tempo do ciclo de varredura do CLP Clic-02 da empresa WEG, testando sua capacidade de se manter constante durante a execução das instruções do programa, ao variar a quantidade de blocos funcionais e o estado das entradas e saídas do CLP. Todo o método foi desenvolvido via o *software* Ladder fornecido pela fabricante do CLP, e os resultados colhidos com o auxílio de um osciloscópio. Após análise e discussão, concluiu-se que o tempo do ciclo de scan varia de forma diretamente proporcional a quantidade de instruções e blocos funcionais inseridos no programa, quase linear, mas respeitando o mesmo tempo durante a execução. Porém, o tempo variou durante a execução de um dos programas, quando testada a função de temporizador, invalidando a condição de tempo real do CLP, e o classificando como tempo quase real. Esta verificação era de se esperar, conforme classificação do próprio CLIC-02, o qual é denominado, pela fabricante WEG, de relé programável.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

A automação industrial, nos moldes atuais, exige a realização de muitas funções, e na Figura 1 pode-se observar os diferentes níveis de automação encontrados em uma planta industrial, em forma de pirâmide (Castrucci, 2007).

A chamada Pirâmide de Automação pode ser descrita brevemente em cada nível por:

Nivel 5: é o nivel responsavel pela administracao dos recursos da empresa, em que se encontram os *softwares* de gestão de vendas e gestão financeira.

Nivel 4: é o nivel responsável pela programação e planejamento da produção realizando o controle e logística dos suprimentos.

Nivel 3: é onde o processo produtivo da planta e controlado. Normalmente, e constituido por bancos de dados com informacoes dos indices de qualidade, relatorios e estatisticas de processo.

Nivel 2: é algum tipo de supervisao associada ao processo. É onde se encontram concentradores de informacao sobre Nivel 1 e as interfaces homem-máquina.

Nível 1: é o nível das máquinas, dispositivos e componentes de chão de fábrica.

É no nível 1 que ocorre o controle das variáveis do processo, através dos CLP's, e é nele que está nosso foco aqui.

Figura 1 – Pirâmide da Automação.

1.1 Controlador Lógico Programável

Segundo a Associação Brasileira de Normas Técnicas – ABNT, CLP é um equipamento eletrônico digital com hardware e software compatíveis com aplicações industriais. E de acordo com a *National Electrical Manufactures Association* – NEMA, é um aparelho eletrônico digital que utiliza uma memória programável para armazenamento interno de instruções para implementações específicas, como lógica, sequenciamento, temporização, contagem e aritmética, para controlar, através de módulos de entradas e saídas, vários tipos de máquinas e processos (Parede, 2011).

De forma geral, um CLP é um computador dedicado a área industrial, constituído por módulos de entradas e saídas (*Hardware*), terminal de programação (*Software*), memórias do tipo fixa e volátil, uma fonte de alimentação e uma Unidade Central de Processamento (do inglês *Central Processing Unit* – CPU), (Martins, 2007). A Figura 2 ilustra os elementos que o CLP é constituído e parte de seu funcionamento em forma de fluxograma.

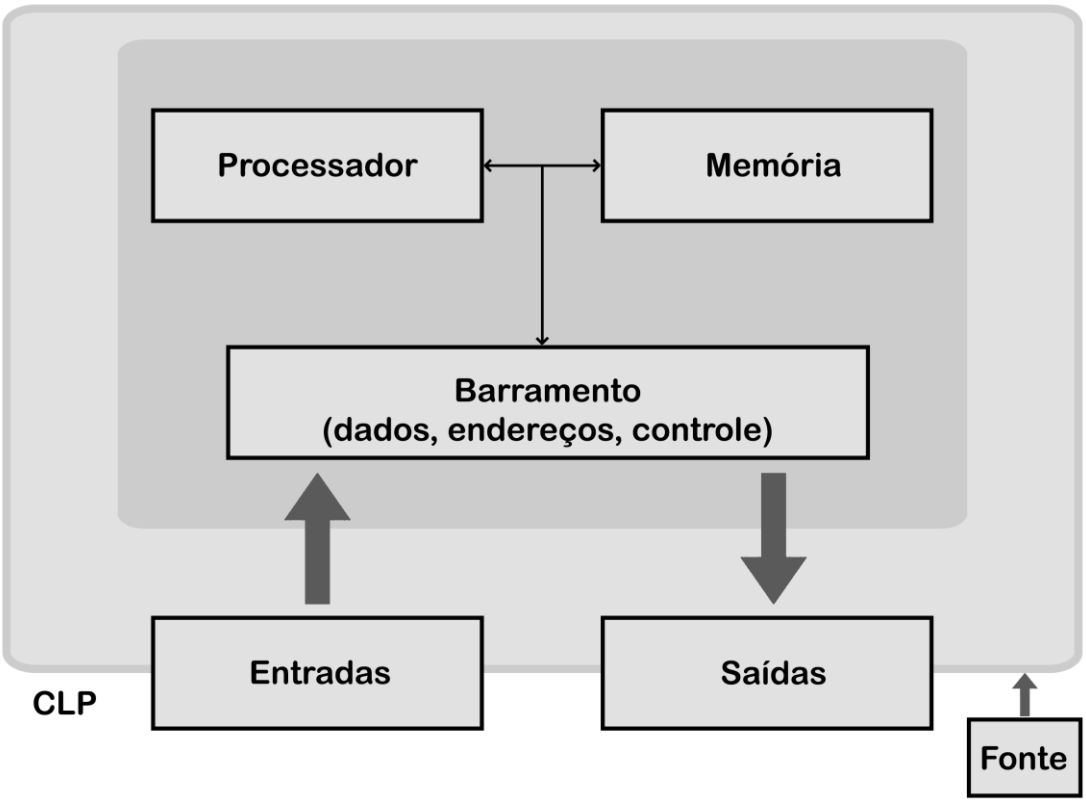


Figura 2 – Elementos de um CLP. (Martins, 2007).

A CPU gerencia todo o sistema, controlando e executando instruções presentes na memória (Parede, 2011), as quais podem ser voláteis e as não-voláteis.

A memória RAM, do inglês *Random Access Memory*, é a memória volátil de acesso aleatório, responsável, dentre outras coisas, por armazenar os valores de entrada e saída do CLP. A memória EEPROM é uma memória não-volátil, que armazena as instruções do programa feito pelo usuário, eletricamente escrita e apagada (Martins, 2007; WEG, 2010).

O Terminal de programação é o local onde o usuário programa as instruções a serem executadas pela CPU do CLP. Normalmente, usa-se um computador pessoal com um *software* dedicado no desenvolvimento desse programa (Martins, 2007).

Os módulos de Entrada e Saída – E/S, também conhecido com I/O do inglês *in* e *out*, são responsáveis pela aquisição de dados de variáveis do processo (entrada) e acionamento de dispositivos físicos (saída) (Martins, 2007).

Basicamente, em um processo de automação industrial, o CLP é responsável por atuar diretamente na planta, sendo controlado por uma Interface Homem-Máquina - IHM, que provê dados para análises em níveis superiores de gerenciamento, como mostrado no fluxograma da Figura 3, porém seu detalhamento foge ao escopo desse trabalho.

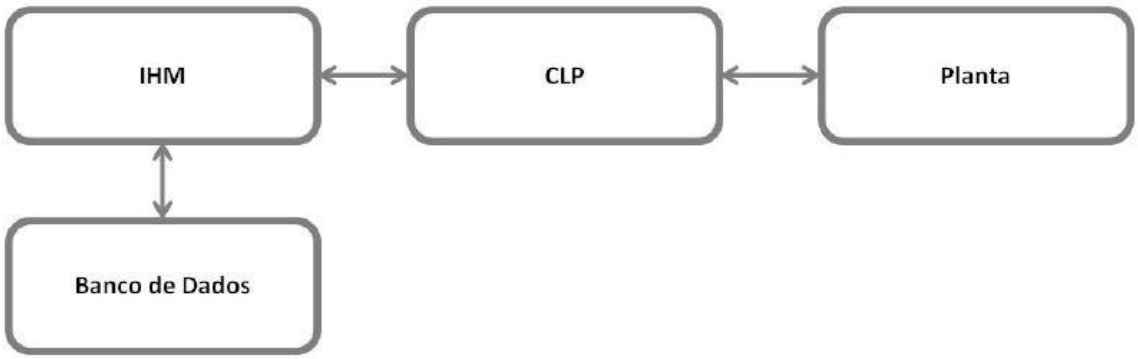


Figura 3 – Fluxograma de informações (Pedruzi, 2014).

Internamente, o CLP quando iniciado executa uma série de verificações pré-programadas, sem acesso do usuário. Após essa inicialização, o CLP entra em um ciclo fechado, executando as seguintes etapas básicas, ilustradas na Figura 4 (Castrucci, 2007):

- Lê e salva na memória os valores de entrada.

- Leitura armazenada em celulas de memória denominada Imagem

do Processo das Entradas (PII).

- Processa as instruções do programa, sequencialmente, em função das entradas.

- Atualiza as saídas de acordo com as instruções e entradas.

- As informações são armazenadas em células de memória denominada Imagem do Processo de Saída (PIO) e transferidas para as saídas do CLP, que atuaram no processo.

Figura 4 – Ciclo de varredura (Castrucci, 2007).

Esse ciclo é conhecido como ciclo de varredura, ou ciclo de *scan*, e se repete continuamente enquanto o CLP estiver operando. É esse ciclo que será estudado nesse trabalho.

Ciclo de *scan*, ou ciclo de varredura, é um ciclo completo de execução do programa contido no CLP, desde a leitura das entradas até a atualização das saídas (WEG, 2010).

Atualmente, os CLP’s trabalham com um tempo pré-definido para esse ciclo, chamado de tempo real, a ser definido posteriormente. Mas antes é preciso compreender a importância desse tempo de varredura.

É fácil observar que a leitura de dados da planta feita pelo CLP não se dá de forma contínua, uma vez que há um tempo entre a leitura das entradas, execução do programa e atualização das saídas, até novamente adquirir os dados do processo. Por isso que a aquisição de dados é feita em tempo discreto, e chegar o mais próximo possível do contínuo é o desejado. Logo, é muito importante que o tempo do ciclo de *scan* seja bem pequeno.

Logicamente, para algumas aplicações esse tempo é mais importante que para outras. Imagine uma planta que controla a temperatura dentro de um alto-forno de ferro guza, em uma usina siderúrgica, Figura 5. Trata-se de uma temperatura de aproxmadamente 1200 ºC e uma planta robusta, em que variações consideráveis não ocorreriam em milissegundos e sua dinâmica é lenta (Gasparini, 2016). Agora imagine a dificuldade de se fazer a contagem de ovos em uma esteira, Figura 6, em que passam vários ovos por segundo. Nessa segunda aplicação, quanto menor o ciclo de varredura, mais confiável será o sistema, uma vez que conseguirá pegar as variações do sensor de presença e de contagem, que serão bem rápidas.

Figura 5 – Alto-forno de indústria siderúrgica.

Figura 6 – Contagem de ovos. (Ovotron, 2018).

Uma situação onde a influência do ciclo de *scan* é importante, é ilustrado na Figura 7. Verifica-se que o ciclo de scan inicia no instante 2ms e finaliza no instante 12ms. Após o início do ciclo de *scan*, ocorreu um evento. Porém esse evento foi finalizado antes do fim do ciclo de varredura, e esse acionamento na entrada não foi e não será detectado pelo ciclo de *scan*, pois o processo de leitura dos sinais de entrada já havia ocorrido. Se isso for associado ao sistema de contagem apresentado anteriormente, o pulso que foi dado durante o ciclo de varredura não será detectado e, consequentemente, a contagem não será efetuada.

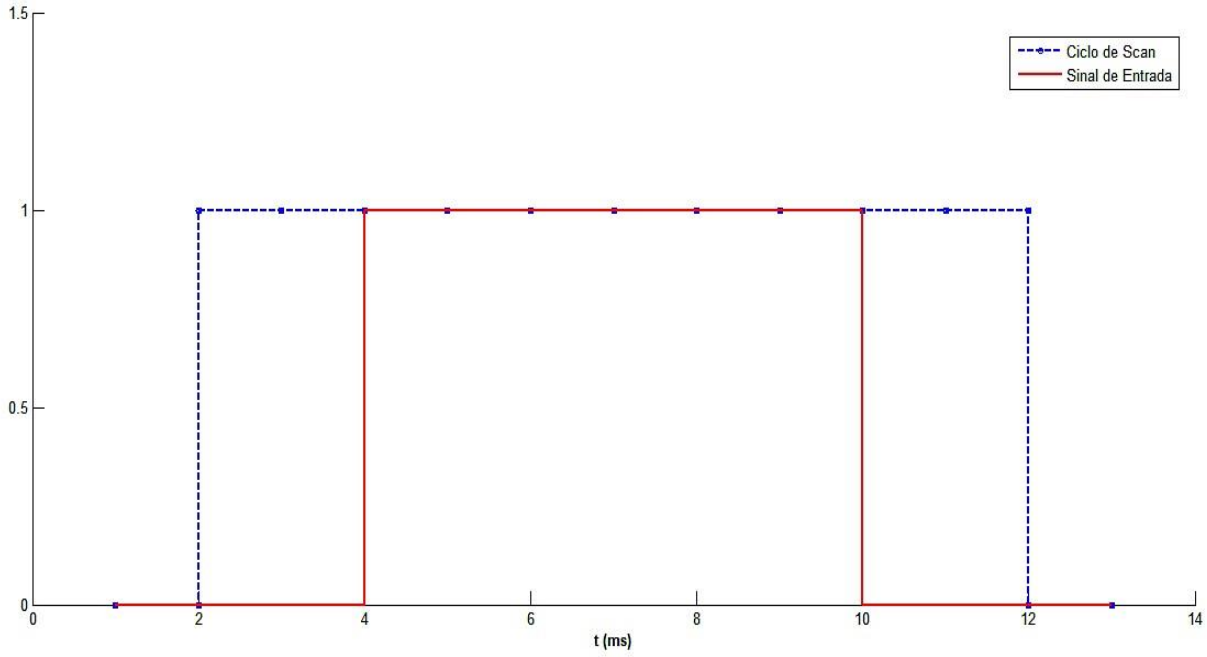


Figura 7 – Ciclo de Scan representado graficamente em função do tempo.

Não deve-se limitar a apenas esses dois exemplos, visto que são infinitas as aplicações. Inclusive questões de segurança são ainda mais importantes do que perder alguns ovos, e por isso é muito importante dar atenção ao tempo de varredura na escolha de um CLP, de acordo com a aplicação.

É precisso definir conceitualmente as diferenças entre um sistema online, um sistema de tempo quase real e um sistema de tempo real. Um sistema diz estar online com outro quando, na mudança de estado de um, o outro é afetado diretamente por esta mudança, estando estes fisicamente conectados ou não. Enquanto um sistema em tempo real é aquele que garante ações e respostas em intervalos de tempo bem definidos, ou seja, após ocorrer uma ação, a próxima é dada por um intervalo fixo de tempo e sempre irá ocorrer nesse mesmo intervalo (Brandão, 2012). Logo, pode-se definir um sistema como quase real semelhante ao de tempo real, mas não tem esse intervalo de tempo bem definido, podendo variar de um ciclo para o outro, dependendo das ações e respostas a instruções do programa.

1.2 Micro Controlador Programável CLIC-02 da WEG

O uso de pequenos CLP's para controlar processos locais ou tarefas simples é uma tendência no mercado atualmente (Martins, 2007). Esses modelos também se tornam bastante interessante no uso didático.

Nesse trabalho, será analisado o ciclo de varredura do Micro-CLP CLIC-02 da WEG, mais especificamente o modelo 20VT-D, semelhante ao da Figura 8. De acordo com o manual da fabricante, WEG, O número 20 diz respeio ao número de entradas e saídas do CLP; 'V' significa que possui LCD, teclado e comunicação RS-485; 'T' refere às saídas, que são do tipo transistor; e 'D' é o tipo de alimentação: contínua de 24V (WEG, 2010).

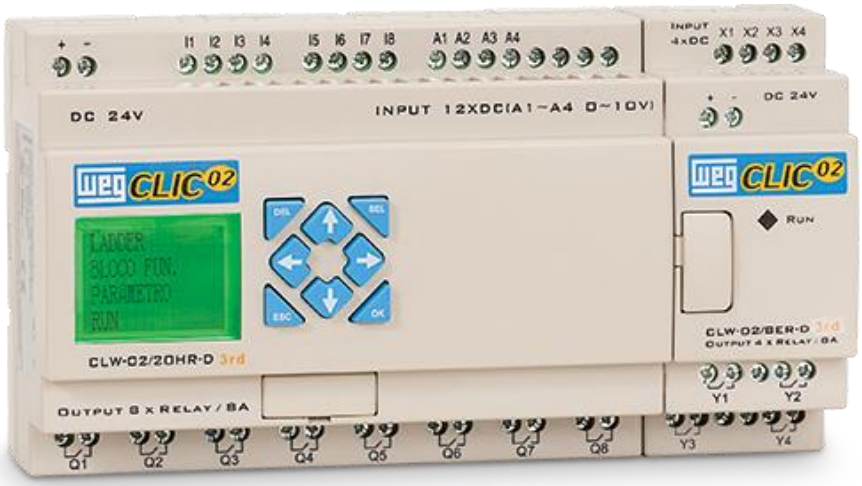


Figura 8 – Micro-CLP CLIC-02 da WEG (WEG, 2010).

É importante atentar para o valor da tensão de entrada, pois o CLP funciona também em valores menores que 24V, mas com recursos limitados e podendo apresentar um desempenho inferior de processamento, o que interferiria diretamente nos nossos resultados. Por isso a necessidade de se manter um valor de entrada sempre constante (WEG, 2010).

As especificações do CLIC-02 pertinentes para o trabalho estão listadas na tabela 1 a seguir:

Tabela 1 – Especificações do CLP Clic-02 (WEG, 2010).

Rede de Alimentação	
Faixa de Tensão de Entrada	20,4 - 28,8 Vcc

Programação	
Linguagens de Programação	Ladder / FBD
Tamanho Máximo do Programa	300 Linhas ou 260 Blocos de Função
Armazenamento do Programa	Memória Flash
Velocidade de Processamento	10ms/ciclo
Quantidade Máxima de Instruções (Temporizadores)	Ladder: 31; FBD: 250
Quantidade Máxima de Instruções (Contadores)	Ladder: 31; FBD: 250

Nela pode-se notar que a velocidade de processamento é de 10ms/ciclo, o principal alvo de estudo desse trabalho.

Há também na tabela as duas formas de programação fornecidas pelo CLIC-02, Ladder e FBD (*Function Block Diagram*), exemplificadas nas Figura 9 e Figura 10 respectivamente, e suas respectivas capacidades. Ladder (que em uma tradução livre para o português significa escada), ou diagrama de escada, é um auxílio gráfico de programação em que as funções lógicas são representadas por contatos de bobinas, facilitando a implementação das instruções de programa no CLP pelo usuário, sempre de forma sequencial. O FBD segue o mesmo conceito gráfico, mas sua representação é feita por blocos de funções. Ambas as linguagens são regulamentadas pela norma IEC-1131, que conta com mais três além dessas duas, que são: texto estruturado, lista de instruções e Diagramas de Funcoes Sequenciais (*Sequential Function Chart* – SFC).

Figura 9 – Bloco de função temporizador no ladder (WEG, 2010).

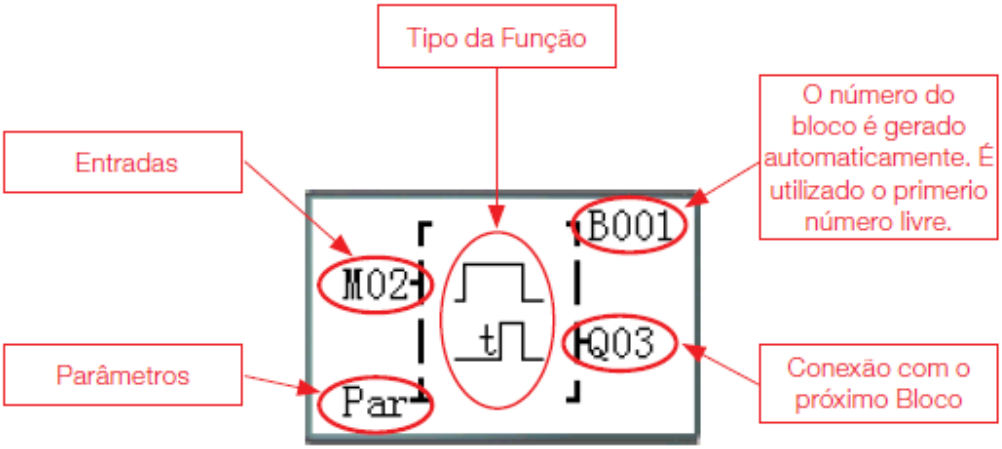
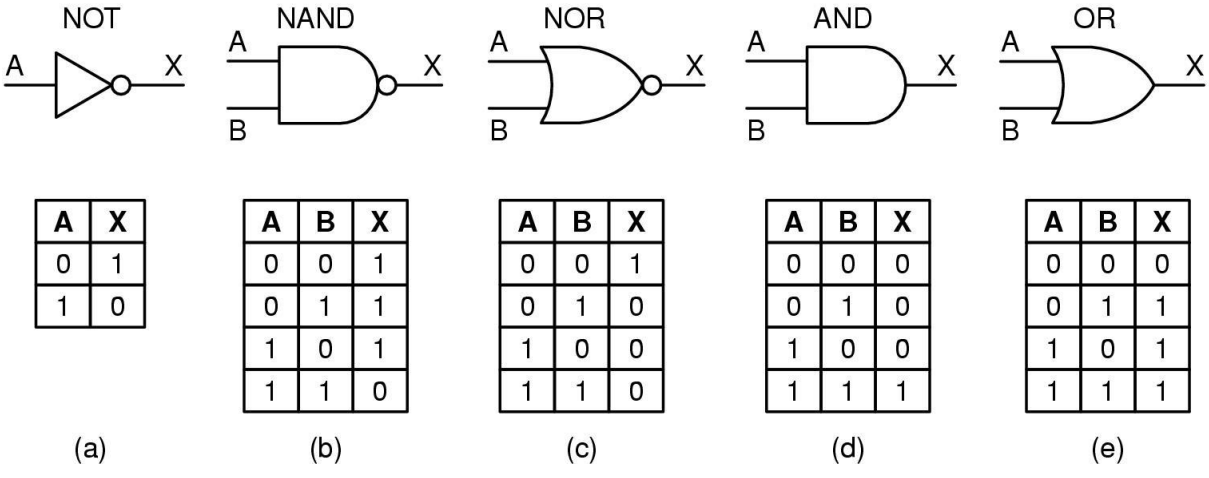


Figura 10 – Bloco de função temporizador no FBD (WEG, 2010).

1.2.1 Funções Básicas

A programação em CLP's fornece várias opções de funções, algumas mais simples e outras mais complexas. Como não é o foco desse trabalho, serão introduzidas apenas as funções consideradas mais básicas e essenciais para o trabalho.

As principais funções são as lógicas: AND, OR, NAND e NOR. Suas respectivas tabelas verdade estão demonstradas a seguir, na Figura 11.



A	X
0	1
1	0

(a)

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

(b)

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

(c)

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

(d)

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

(e)

Figura 11 – Tabelas verdade das funções lógicas (a) AND (b) OR (c) NAND (d) NOR.

Além dessas funções lógicas, algumas instruções contribuem para escrever a lógica dos programas, como bordas de subida e descida, que detectam somente o pulso na subida ou

na descida do sinal, como na Figura 12. Outra instrução muito útil é o SET e o RESET, que trava uma variável de saída em estado alto, até que o comando de RESET é dado.

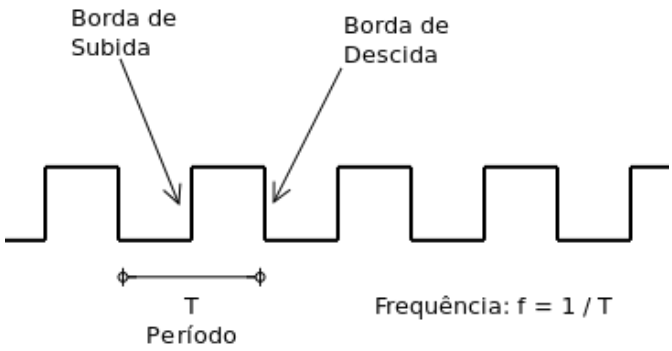


Figura 12 – Bordas de subida e descida.

Ainda tem blocos funcionais importantes, como temporizador, comparador e contador, os quais serão exemplificados na sequência.

1.2.2 Funções Especiais

a) Temporizador com retardo na energização

Existem vários modos de operação para o temporizador, como retardo na energização e retardo na desenergização. O primeiro citado (modo 1), um dos mais usados, será analisado nesse trabalho.

De acordo com um valor de tempo t escolhido pelo usuário, a saída do temporizador é habilitada após a entrada ficar em estado de alta durante esse tempo. A saída volta para zero com a entrada em zero também (WEG, 2010). A Figura 13 ilustra graficamente o funcionamento desse modo de operação do temporizador.

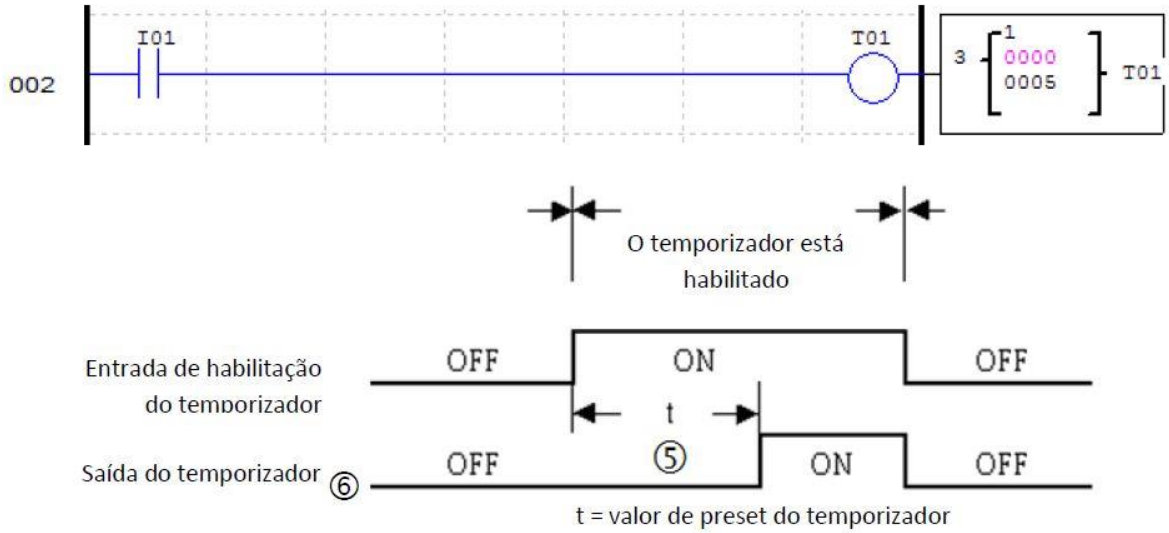


Figura 13 – Temporizador modo 1 – Retardo na energização (WEG, 2010).

b) Contador de contagem fixa e não-retentiva

A exemplo do temporizador, o contador também apresenta diferentes modos de operação. Como não é o foco desse trabalho explicar todos, somente ao modo 1 do CLIC-02 será apresentado.

Quando em contagem no modo crescente, os pulsos de contagem irão incrementar o valor de contagem até atingir o valor previamente escolhido pelo usuário, ligando assim os contatos de saída do contator. Para desligar é necessário acionar a entrada de *reset* ou mudar a direção da contagem (WEG, 2010). Seu funcionamento está representado na Figura 14.

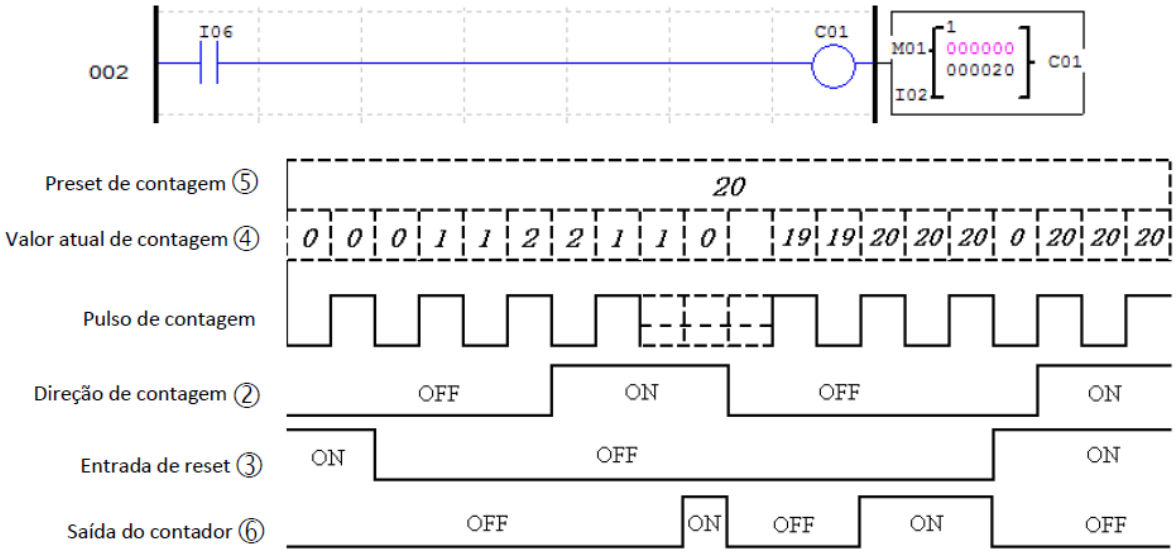


Figura 14 – Contador modo 1 – contagem fixa e não-retentiva (WEG, 2010).

c) Comparador

Os comparadores, como o nome já diz, compara o valor lido, de uma variável de entrada analógica, com um valor de referência, realizando a operação desejada pelo usuário. Essa operação pode ser uma simples comparação, de que quando o valor de entrada é menor que o valor de referência, a saída é zero, e quando é maior, a saída fica em estado alto. Ou pode ser outras operações, como na Tabela 2 a seguir (WEG, 2010).

Tabela 2 – Modos de operação de um bloco comparador (WEG, 2010).

Modo 1	Ay - Ref ≤ Ax ≤ Ay + Ref
Modo 2	Ax ≤ Ay
Modo 3	Ax ≥ Ay
Modo 4	Ref ≥ Ax
Modo 5	Ref ≤ Ax
Modo 6	Ref = Ax
Modo 7	Ref ≠ Ax

1.3 Objetivo Geral

Este trabalho tem como objetivo coletar dados para identificar e validar a influência da inserção de blocos funcionais, no programa Ladder, na variação do tempo de ciclo de varredura do CLP CLIC-02, principalmente durante sua execução.

Dado o objetivo geral, têm-se como objetivos específicos:

- Analisar a variação do tempo de ciclo de varredura com a inserção de funções lógicas no programa Ladder, com contatos normalmente aberto e normalmente fechado;
- Analisar a variação do tempo de ciclo de varredura com a inserção de blocos funcionais do tipo temporizador no modo 1, também no programa Ladder, desligado e ligando; e,
- Analisar a variação do tempo de ciclo de varredura com a inserção de blocos funcionais do tipo contador no modo 1, também no programa Ladder, desligado, ligado e contando.

1.4 Estrutura do Trabalho

Este trabalho está estruturado em 4 capítulos. No primeiro capítulo foi abordado uma breve introdução sobre automação e o que motiva a realização desse trabalho. O leitor também foi apresentado ao CLP CLIC-02 e suas funções, bem como aplicações de CLP e os objetivos desse trabalho, além da introdução de conceitos pertinentes para o entendimento do mesmo, como ciclo de varredura e definições de tempo online, real e quase real.

No Capítulo 2, serão apresentados os materiais e a metodologia utilizada para alcançar os objetivos propostos, desde a montagem física com osciloscópio e CLP, até a elaboração e execução dos programas implementados no controlador lógico programável.

No Capítulo 3, são mostrados os resultados obtidos pela análise da inserção de cada bloco no programa do CLP. São esses os blocos: portas lógicas AND e OR, Temporizador e Contador. Com esses resultados, fica possível estimar o tempo de cada ciclo de *scan* a depender do programa inserido.

Portanto, fica para o Capítulo 4, as conclusões a cerca dos resultados obtidos, com destaque para o objetivo principal alcançado: provar que o CLP varia seu ciclo de varredura durante a execução do programa, a depender de uma função especial, o temporizador.

2 Materiais e Métodos

Para coletar os dados do CLP e analisar seu tempo de ciclo de *scan*, foram utilizados, no Laboratório de Automação do Departamento de Engenharia Elétrica da Universidade Federal de Viçosa, os seguinte equipamentos e materiais:

- CLP CLIC-02 20VT-D;
- Osciloscópio Tektronix TDS 1012 de dois canais;
- protoboard;
- fonte de tensão 18v cc;
- computador pessoal; e,
- diodo led.

Além desses materiais físicos, foi utilizado também o *software* fornecido pela WEG para programar em Ladder ou FBD para o CLP CLIC-02, o “Clic02 Edit”. A Figura 15 ilustra o local utilizado para coleta de dados, com os respectivos equipamentos.

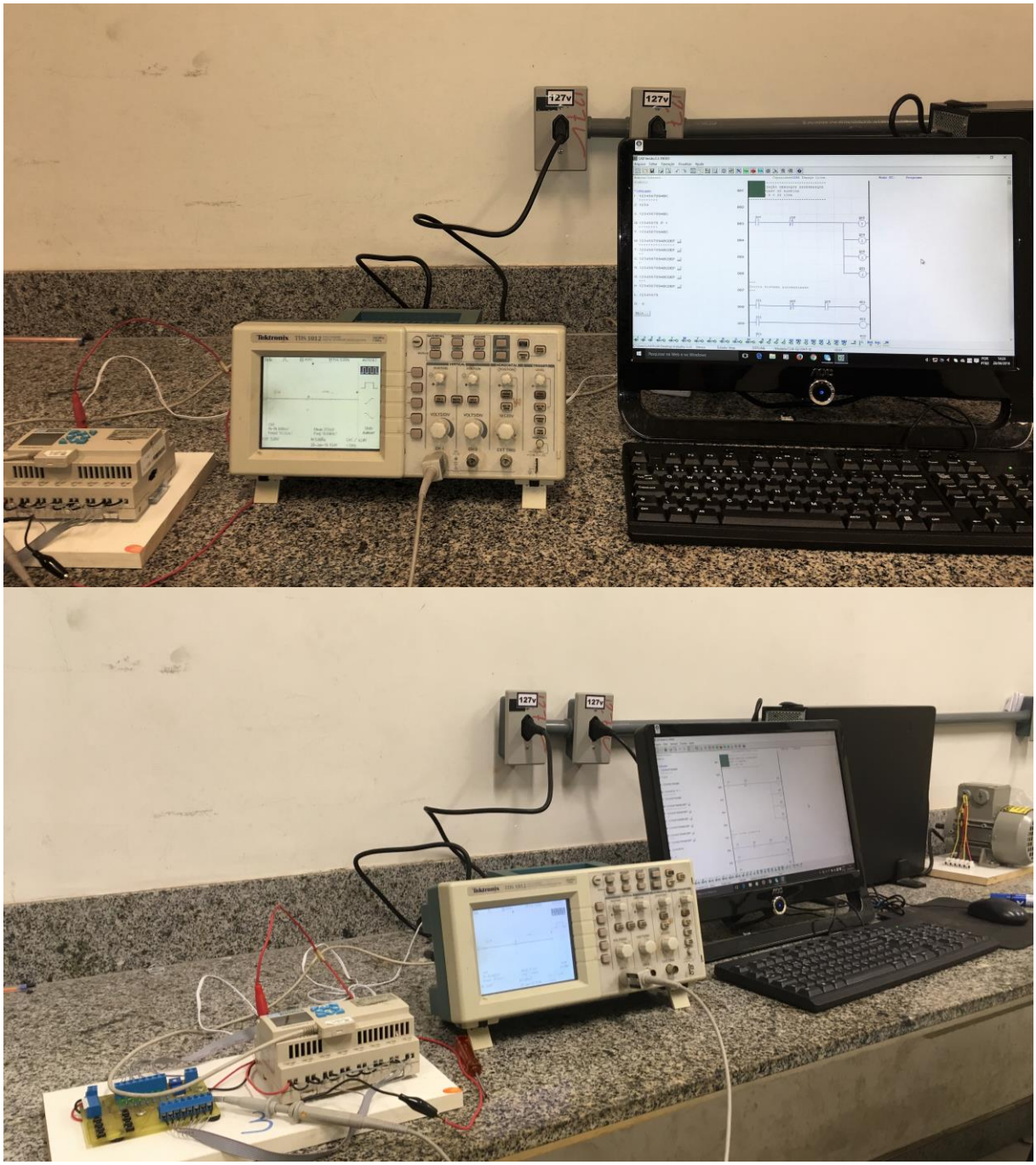


Figura 15 – Materiais utilizados devidamente montados para os experimentos.

O computador pessoal foi utilizado para escrever as instruções dos programas implementados no CLP, para fim de analisar o tempo do ciclo de varredura de acordo com cada programa. A linguagem escolhida foi a Ladder, por ser mais didática.

Uma fonte de tensão de 18V cc alimentou o CLP e a saída Q1 do mesmo. A saída Q1 foi a escolhida pois é uma saída transistorizada de alta frequência do CLP (WEG, 2010)., visto que era esperado uma frequência alta do ciclo de *scan*. Essa saída precisa ser alimentada

em um lado, para o CLP fechar o contato da saída, de acordo com as instruções do programa, e assim a energia fluir para o outro lado, como esquematizado na Figura 16. Para conferir visualmente se a execução virtual do programa está sendo aplicada fisicamente à saída do CLP, foi colocado uma luz LED em seu terminal.

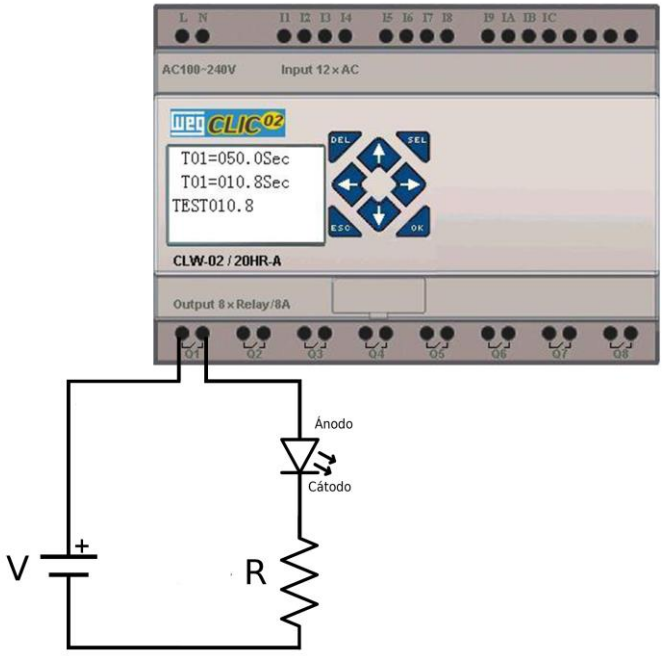


Figura 16 – Esquema elétrico utilizado.

Por fim, o osciloscópio foi conectado na saída Q1, devidamente aterrado, para uma coleta de dados precisa. Dessa forma, fica fácil anotar o período de cada ciclo estudado.

2.1 Detecção do Ciclo de Scan

Uma simples instrução foi adicionada no programa para verificar o tempo do ciclo de varredura do CLP. Nenhuma entrada foi utilizada, apenas a saída Q1. Lembrando que as instruções do programa são lidas de forma sequencial, logo, essa foi colocada no início. Ela consiste em basicamente implementar o bloco de Q1 barrado (normalmente fechado), acionando a própria saída Q1, como na Figura 17. Perceba que com essa manobra, a saída irá alternar entre os sinais de alta e baixa a cada ciclo, formando uma onda quadrada em que meio período é o equivalente a um ciclo de varredura, ilustrado na Figura 18.

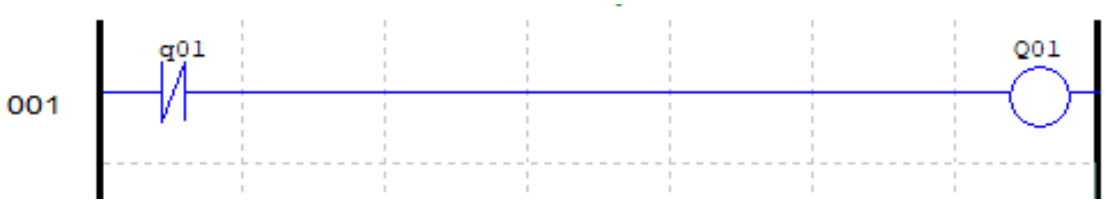


Figura 17 – Método para medir o ciclo de *scan* (WEG, 2010).

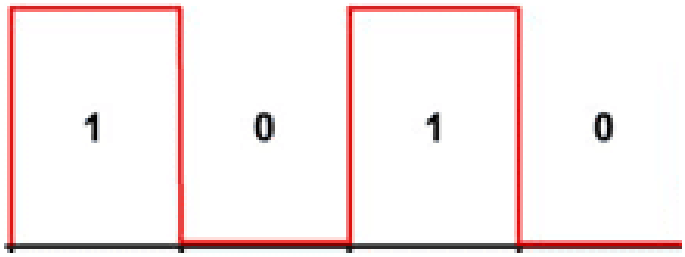


Figura 18 – Meio período equivale a 1 ciclo de *scan* (WEG, 2010).

2.1 Implementando Funções

Uma vez que se tem o ciclo de *scan* definido sem instruções adicionais, os dados serão coletados implementando algumas funções. Espera-se que o ciclo aumente diretamente proporcional ao incremento de linhas de programa.

2.2.1 Funções Lógicas AND, OR, NAND e NOR

As primeiras funções incrementadas foram as lógicas. Primeiro com AND (Figura 19), depois NAND (Figura 20), OR (Figura 21), e o mesmo se repetiu com AND e OR juntas (Figura 22), sempre começando com um bloco somente e indo até altos valores perto ou no máximo da capacidade do programa.

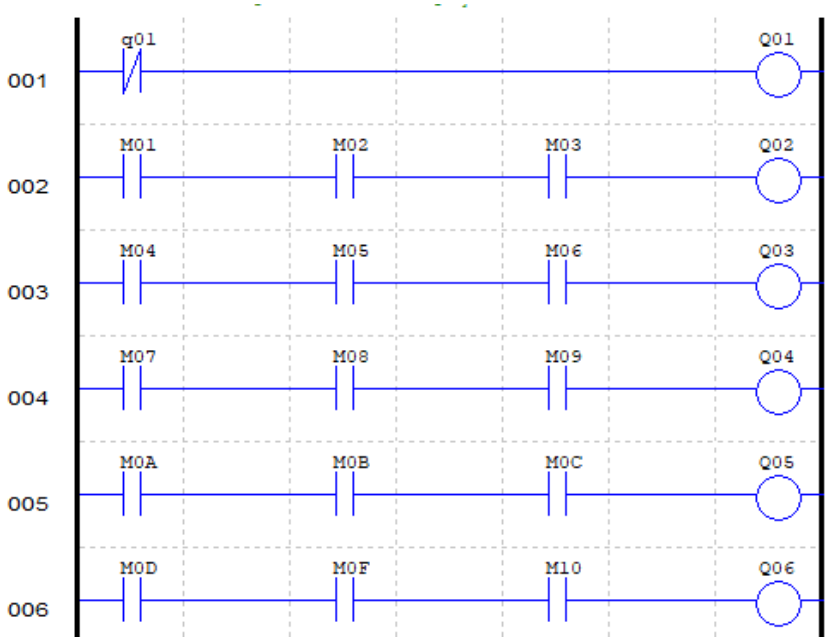


Figura 19 – Blocos AND no programa Ladder.

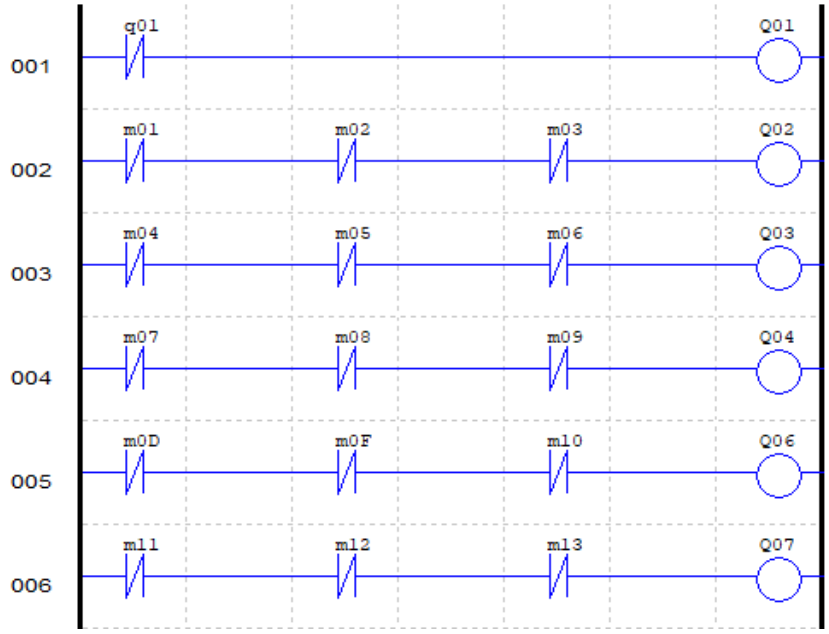


Figura 20 – Blocos NAND no programa Ladder.

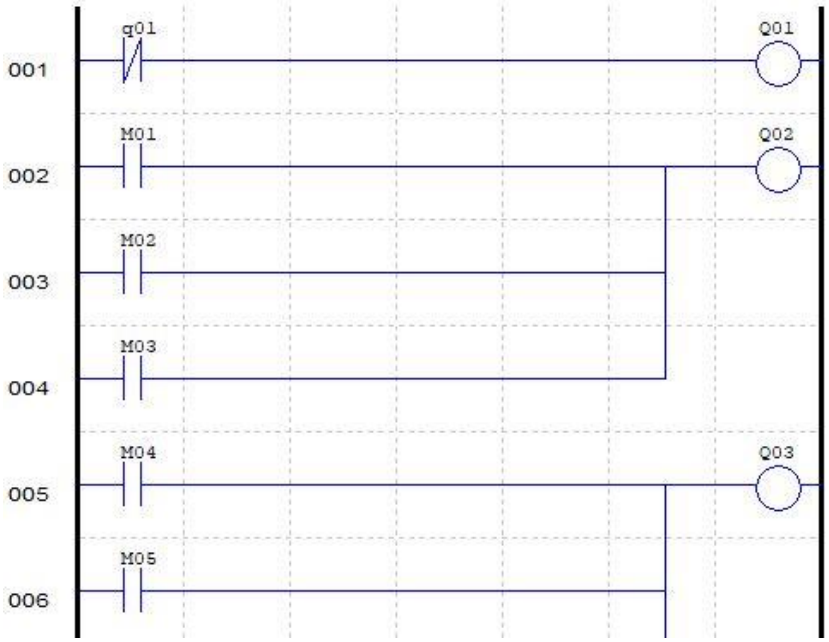


Figura 21 – Blocos OR no programa Ladder.

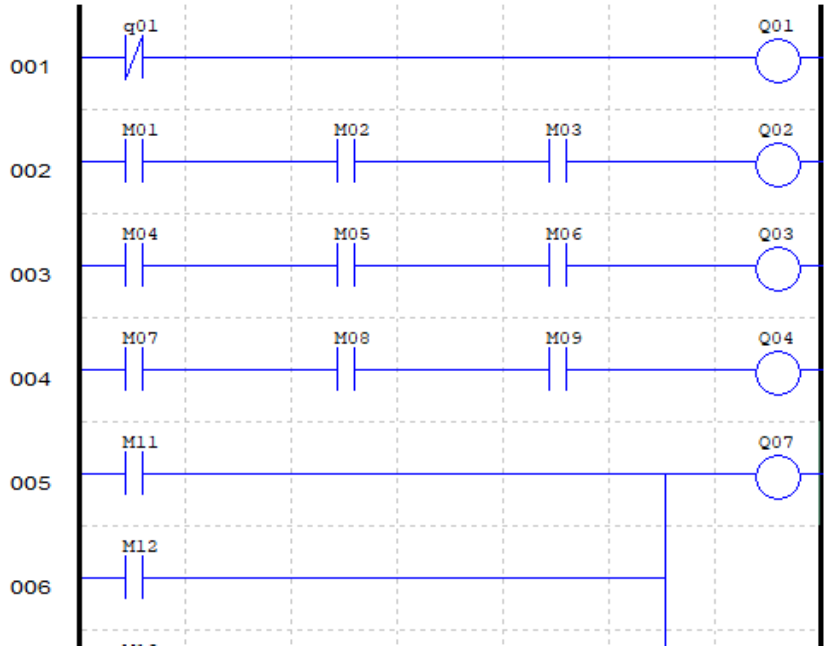


Figura 22 – Blocos AND e OR no programa Ladder.

Para o experimento, foram utilizadas variáveis auxiliares (M e N no Clic02), e quando essas não foram suficientes, o programa em Ladder permite utilizar outros blocos funcionais no modo 0, que é um modo de contato de bobina, i.e., atua simplesmente como variável auxiliar.

Na busca de mensurar a duração do ciclo de varredura em relação ao tamanho e às funções do programa, foram utilizadas diferentes saídas para cada função lógica, e também a mesma saída para todas.

Além disso, o computador pessoal foi conectado ao CLP através da porta serial RS232, e as variáveis auxiliares foram acionadas, acionando também as saídas, com o objetivo de perceber alguma variação no tempo do ciclo de varredura durante a execução do programa e alteração nos estados dessas variáveis.

2.2.2 Temporizador

Como já foi citado anteriormente, o temporizador do CLIC-02 pode operar em diferentes modos, dependendo da aplicação. Os experimentos serão feitos com o temporizador no modo 1, que é o de retardo na energização, já explicado no item a) da seção 1.2.2. Na Figura 23 está exemplificado o programa montado para teste dos temporizadores, com esses em paralelo, uma vez que não faz sentido implementá-los de forma serial.

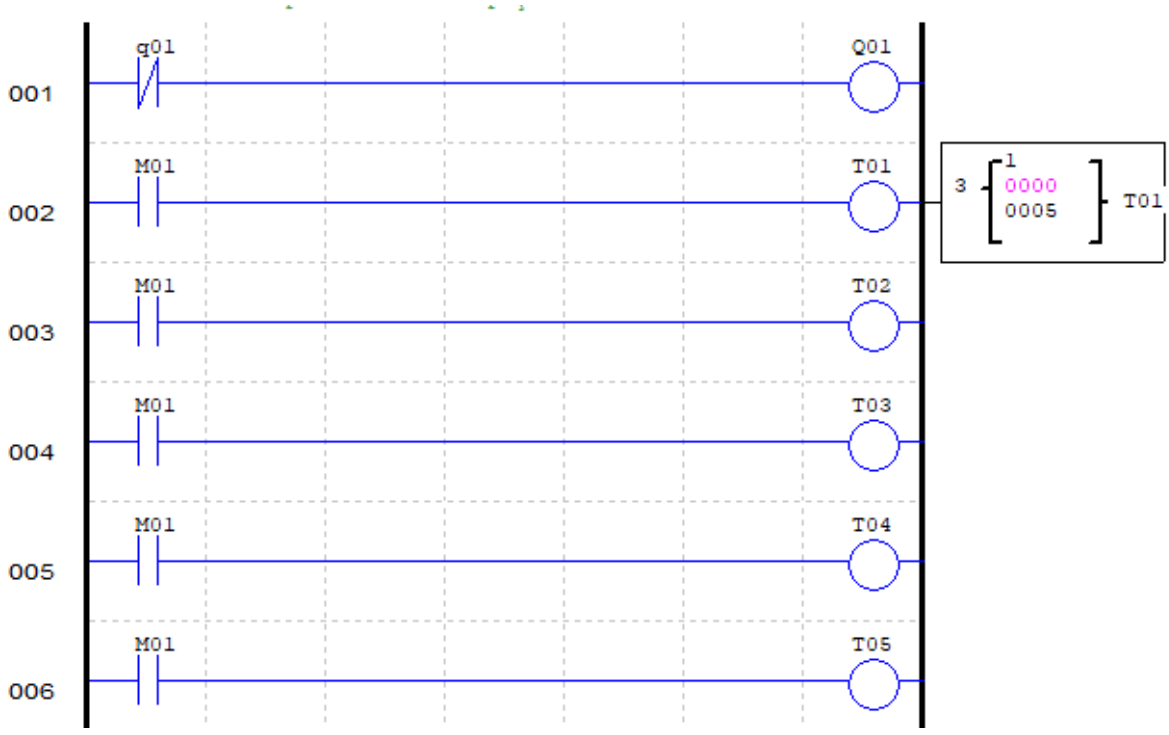


Figura 23 – Blocos Temporizadores no programa Ladder.

Foram utilizados os mesmos métodos de variações nas saídas e variáveis auxiliares das funções lógicas, com o aditivo de acionar o temporizador através da conexão modbus e de

uma entrada física do CLP. A quantidade de blocos temporizadores implementados foram 30, respeitando a capacidade máxima de 31 temporizadores por programa na linguagem Ladder.

2.2.2 Contador

O contador segue a mesma linha do temporizador, pois também opera em diferentes modos e foi utilizado o modo 1, já citado nesse trabalho. A Figura 23 apresenta um exemplo do programa implementado para testar os contadores, também em paralelo, visto que não faz sentido para esse trabalho testá-los em série.

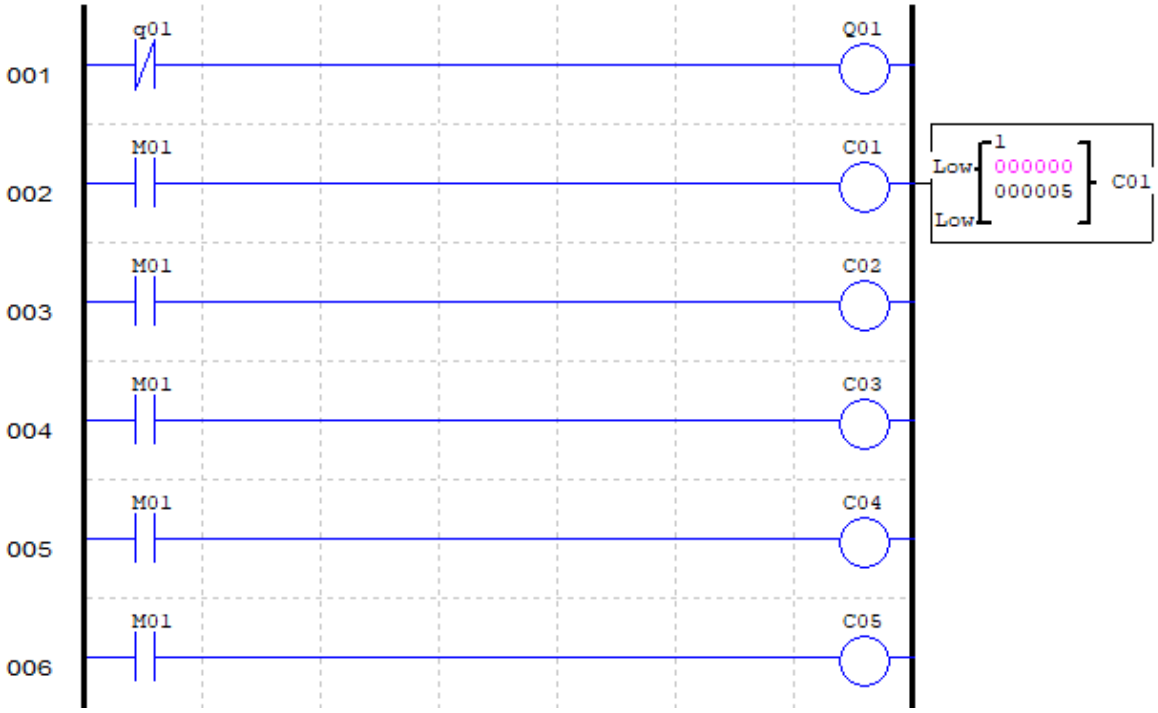


Figura 24 - Blocos contadores no programa Ladder.

Foram utilizados os mesmos métodos de variações nas saídas e variáveis auxiliares anteriores, com o aditivo de acionar o contador através da conexão serial RS232 e de uma entrada física do CLP. A quantidade máxima de blocos contadores implementados foram 30, respeitando a capacidade máxima de 31 contadores por programa na linguagem Ladder.

3 Resultados e Discussões

Uma vez definidos os objetivos e a metodologia, os resultados foram consequência de uma série de testes repetitivos. Não foi preciso tirar a média das repetitivas medições, pois todas apresentaram os mesmos valores nas diferentes aquisições de dados.

Vale lembrar que o tempo do ciclo de *scan* fornecido pelo fabricante é de 10ms, especificado na Tabela 1.

3.1 Funções Lógicas

As tabelas 3 e 4 mostram os valores do ciclo de varredura com a implementação de variáveis auxiliares para a formação de funções lógicas AND e OR, respectivamente. A Figura 25 apresenta esses valores em forma de gráfico, para efeito de comparação visual e melhor compreensão dos resultados.

Tabela 3 – Tempo do ciclo de scan com portas lógicas AND.

Blocos AND	0	1	3	5	10	20	30	40	50
Tempo (ms)	3,11	3,12	3,14	3,16	3,2	3,29	3,38	3,46	3,54

Tabela 4 – Tempo do ciclo de scan com portas lógicas OR.

Blocos OR	0	1	3	5	10	20	30	40	50
Tempo (ms)	3,11	3,12	3,14	3,16	3,21	3,3	3,4	3,5	3,58

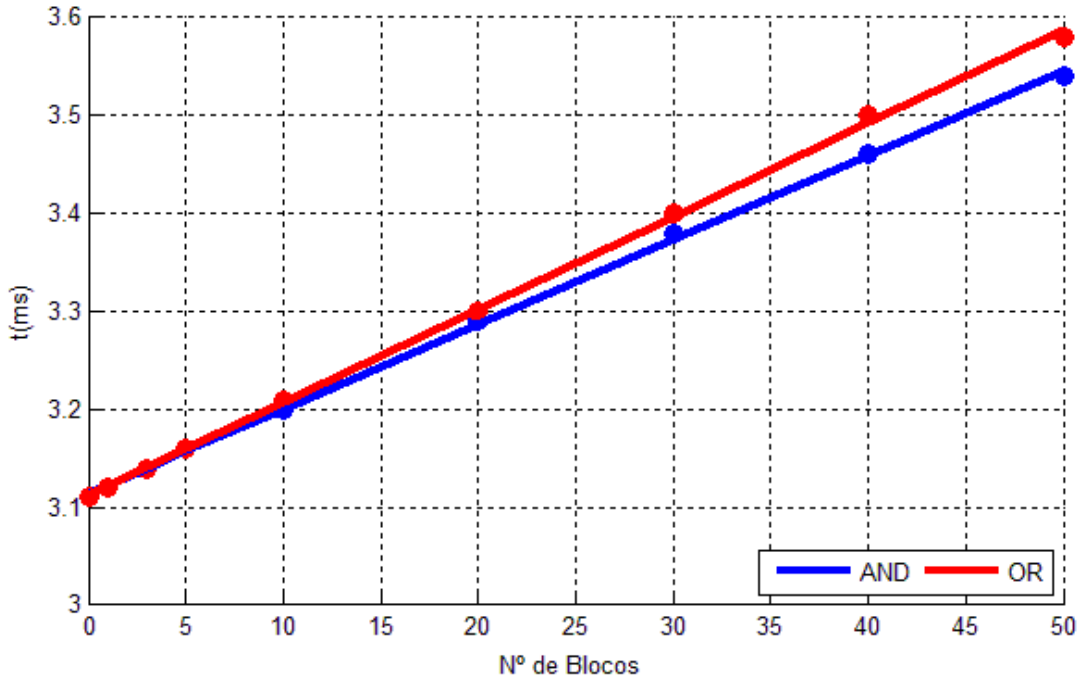


Figura 25 - Gráfico comparativo entre os blocos AND e OR.

Com os valores medidos, foi feita uma regressão linear dos pontos, e a equação (1) pode ser usada para estimar o tempo de duração de um programa com blocos AND e/ou OR, com um coeficiente de determinação de 99,94%.

$$T_b \approx 3{,}11 + 0{,}009.m_{AND} + 0{,}01.n_{OR} \qquad (1)$$

Em que *m* é a quantidade de AND e *n* a quantidade de OR.

Os resultados foram idênticos utilizando saídas iguais ou diferentes, e acionando as variáveis auxiliareis para alterar o estado das saídas ou não.

Mas alguns padrões puderam ser identificados analisando essas tabelas. São esses:

- O tempo de ciclo de *scan* tem um valor inicial fixo, e segue um crescimento quase linear com a adição das portas lógicas nas instruções, tanto AND quanto OR, tendo a duração de aproximadamente 0,01ms cada bloco adicionado.

- A quantidade de linhas e de variáveis no programa são as responsáveis por determinar o tempo de ciclo de *scan.*

- Os programas com porta lógica OR se mostraram um pouco mais lentos a medida que mais instruções foram implementadas. Isso se deve à configuração das variáveis para formar as portas lógicas AND e OR, uma vez que para formar a porta OR se usam mais linhas que para formar as portas lógicas AND.

- A partir disso e de outros experimentos feitos durante o processo, nota-se que a quantidade de linhas é menos importante que a quantidade de variáveis auxiliares na determinação do tempo de ciclo de *scan*, visto que só foi possível observar uma diferença de microssegundos entre as funções lógicas AND e OR com números altos de portas lógicas incrementadas. Isso porque OR usa o triplo de linhas que AND, pois duas portas AND são implementadas em uma linha, enquanto duas OR usam 3 linhas, como mostrado nas Figura 19 e Figura 21.

A tabela 5 confirma os resultados individuais de cada porta lógica anterior.

Tabela 5 – Tempo do ciclo de scan com portas lógicas AND e OR.

AND/OR	0	1/1	3/3	10/10	20/20	40/40	180/146
Tempo (ms)	3,11	3,14	3,18	3,3	3,48	3,88	6,26

Como pode-se observar, ainda há uma relação quase linear entre o tempo de ciclo de varredura e o número de instruções, que poderia ter sido estimada com a equação (1).

Um resultado interessante e diferente dos anteriores, observado no último programa feito, com 180 blocos AND e 146 OR, que foi o máximo possível com 300 linhas, é que o tempo de ciclo de *scan* não ultrapassa o valor dado pelo fabricante de 10ms com essas instruções.

Os resultados de NAND e NOR foram exatamente iguais de AND e OR, dispensando assim a apresentação de seus respectivos valores aqui. Isso significa que o tipo de contato, normalmente fechado ou normalmente aberto, não influencia na duração da leitura do programa. A aplicação de bordas de subida e descida também não provocaram mudanças.

3.2 Temporizador

A tabela 6 apresenta os valores coletados na análise do tempo do ciclo de *scan* com programas em que foram implementados o temporizador de modo 1 – retardo na energização. A Figura 26 ilustra esses dados em forma de gráfico para melhor análise.

Tabela 6 – Duração do ciclo de varredura com temporizador de retardo na energização.

TEMP.	0	1	3	5	10	20	30	Tempo
OFF	3,11	3,12	3,14	3,18	3,24	3,38	3,5	(ms)
ON	3,11	3,14	3,18	3.22	3.34	3.58	3.82	(ms)
Metade ON	-	-	-	-	3,3	3,46	3,66	(ms)

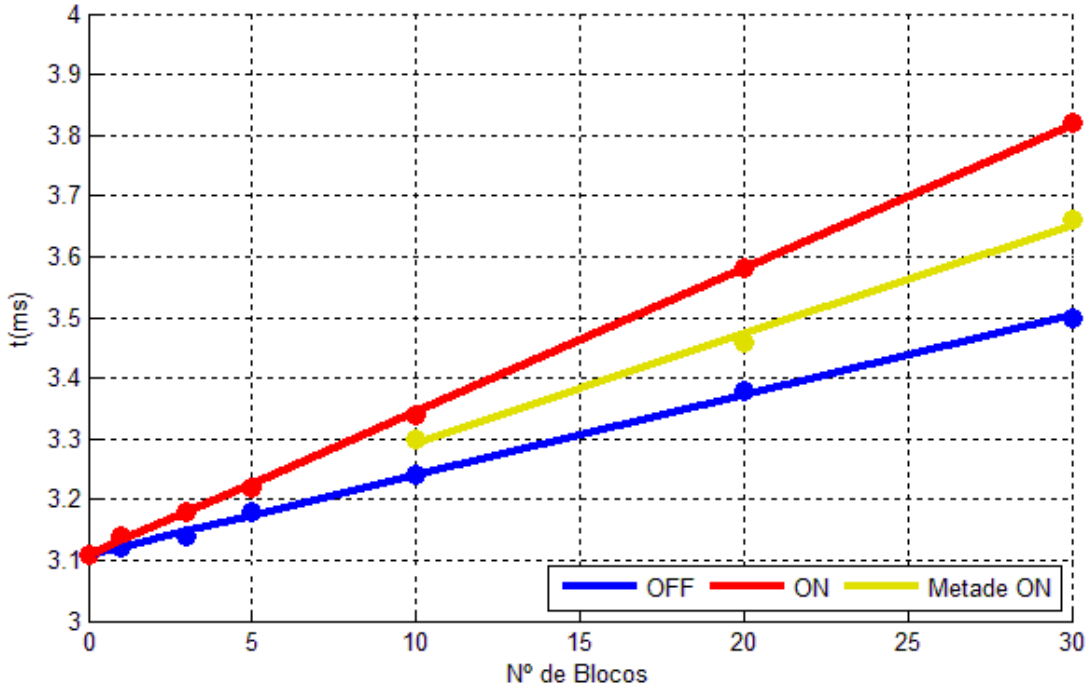


Figura 26 - Gráfico comparativo entre os blocos temporizadores desligados e ligados.

O temporizador foi responsável pelo resultado mais interessante, alcançando o objetivo desse trabalho. É possível perceber que, com o temporizador desligado, o tempo do ciclo de *scan* é parecido com das instruções lógicas. Mas quando o temporizador é acionado, o ciclo de varredura aumenta sua parcela de tempo variável (além dos 3,11ms de quando o programa não tinha instruções).

Neste trabalho não foram analisados os comparadores, mas é importante dizer que os temporizadores têm, dentro deles, um comparador. Isso acontece, pois, a tarefa do

temporizador é de comparação: se o tempo for maior que o pré-determinado pelo usuário, uma saída é acionada, ou seja, dois eventos ocorrem (comparação e acionamento), por isso o tempo do ciclo de varredura aumenta quando o temporizador começa a contar.

Para comprovar esse efeito causado pelo acionamento do temporizador, apenas metade dos temporizadores foram ligados, e foi possível perceber a linearidade no acréscimo de tempo ao ciclo de *scan* provocado pela adição dos blocos funcionais e pelos temporizadores sendo ligados.

Com os valores medidos, foi feita uma regressão linear dos pontos, e a equação (2) pode ser usada para estimar o tempo de duração de um programa com blocos temporizadores modo 1, com um coeficiente de determinação de 99,9%.

$$T_t \approx 3{,}11 + 0{,}013.t_{OFF} + 0{,}01.t_{ON} \qquad (2)$$

Perceba que se há um temporizador acionado, esse deve ser substituído tanto no OFF quanto no ON. O *t* de OFF é referente a simples existência do temporizador no programa, enquanto ON é com ele sendo acionado.

Ficou evidente que o ciclo de varredura variou à medida que os temporizadores foram acionados, i.e., quando acionado, o temporizador opera também como um comparador, fazendo com que alterasse o tempo de ciclo de varredura. E isso aconteceu durante a execução do programa, ou seja, o processo não foi interrompido, instruções novamente gravadas, e CLP religado. Foi entre um ciclo e outro com o CLP ligado.

3.3 Contador

A tabela 7 mostra os valores do tempo de ciclo de scan quando implementados blocos funcionais contadores de contagem fixa e não-retentiva, com o valor de 5 escolhido. A Figura 26 ilustra esses dados em forma de gráfico para melhor análise.

Tabela 7 – Duração do ciclo de scan do CLP Clic-02 com a implementação de contadores modo 1.

Contador	0	1	3	5	10	20	30
Tempo (ms)	3,11	3,14	3,18	3,22	3,34	3,54	3,76

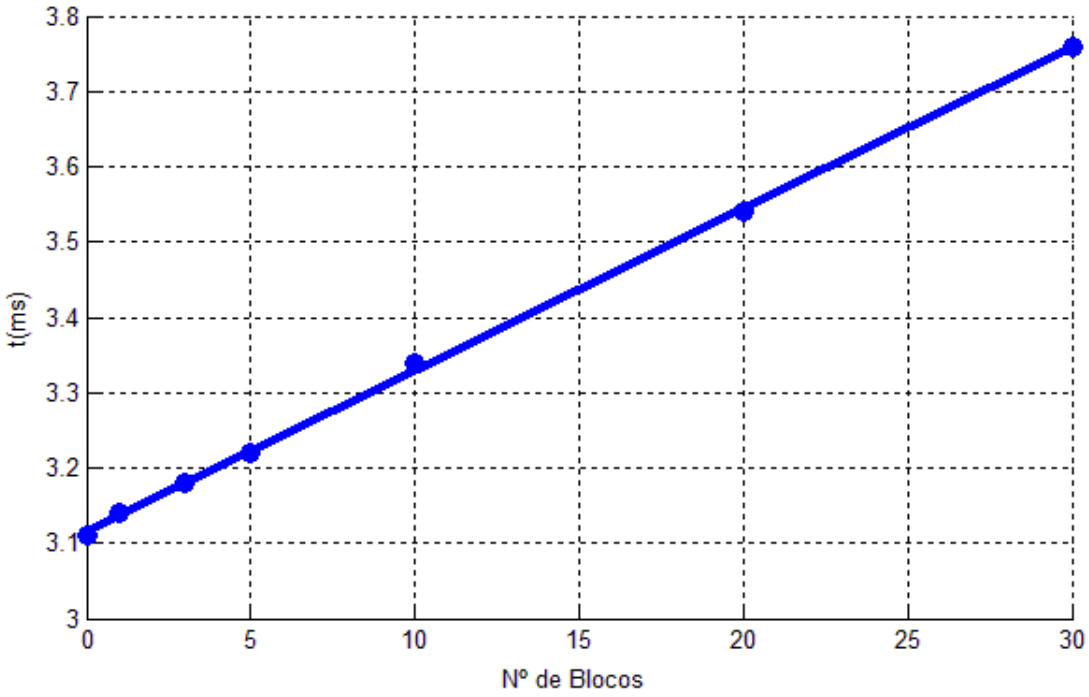


Figura 27 - Gráfico mostrando a linearidade obtida com os blocos contadores.

Com os valores medidos, foi feita uma regressão linear dos pontos, e a equação (3) pode ser usada para estimar o tempo de duração de um programa com blocos contadores modo 1, com um coeficiente de determinação de 99,95%.

$$T_c \approx 3{,}11 + 0{,}021.n_c \qquad (3)$$

Pode-se perceber que os contadores tornam o programa mais lento do que a mesma quantidade de funções lógicas e que os temporizadores desligados. Isso acontece porque o bloco contador também exerce uma comparação interna, visto que ele tem que comparar se o valor da entrada atingiu o valor pré-determinado pelo usuário.

Porém, durante a contagem, feita tanto de forma física quanto pelo computador pessoal em comunicação com o CLP, não houve nenhuma alteração no tempo do ciclo de varredura, i.e., o contador não altera o tempo do ciclo de varredura quando acionado, pois esse não depende de um clock interno do CLP, depende apenas de quem está efetuando o sinal de entrada. Isso explica porque ele torna o programa mais lento que os outros, mas não altera o tempo quando acionado.

A única variação vista no tempo do ciclo de *scan* em todos os programas implementados foi de 0,02ms para mais ou para menos, que pode ser atribuída a ruídos e imperfeições do sistema.

4 Conclusões

Primeiramente, a variação no tempo de ciclo de varredura enquanto o CLP estava ligado e executando as instruções do programa, desconstrói a premissa que o CLP CLIC-02 da WEG opera em tempo real em todos os seus modos de operação, e o classifica como operação em tempo quase real.

Outro fato observado foi que o aumento no tempo de ciclo de *scan* aumenta linearmente com a implementação de instruções ao programa, influenciado principalmente pelo número de variáveis, mas também pela quantidade de linhas do programa. Cada função adiciona um intervalo de tempo ao ciclo, dependendo de qual foi implementada. Pois, como foi visto, AND e OR exigem menos tempo do ciclo que o contador, e esse quase o mesmo que o temporizador acionado.

Conhecer o tempo do ciclo de varredura é muito importante quando o CLP é utilizado para situações em que as informações de sensores são captadas em pequenas frações de segundos. A rigidez desse tempo é essencial para aplicações que requerem alta confiabilidade na leitura das entradas, o que torna o CLIC-02 um equipamento não recomendado para este tipo de cenário, especialmente no uso do temporizador.

Além disso, partindo do princípio que os controladores lógicos programáveis trabalham em tempo real, o CLP Clic-02 pode ser classificado como um relé programável, ou relé inteligente, nomes que o próprio fabricante atribui ao seu produto em seu manual e seu site (CLIC02, 2018; WEG, 2010).

Este trabalho alcançou o objetivo de avaliar o tempo de execução do ciclo de *scan* do CLP CLIC-02 da WEG, e verificar a influência da inserção de blocos funcionais na variação desse tempo, bem como validar a suspeita de que não opera em tempo real como os controladores. Vide, por exemplo, o CLP PLC300 da própria WEG, que, mesmo quando não tem instruções, ele varre as linhas vazias de programa, ocasionando em tempo ocioso de ciclo de scan (alguns apresentam escolha para o usuário entre o scan mais rápido possível e ciclo de scan constante), (PLC300, 2017; TPW, 2017).

Com as fórmulas de tempo apresentadas na seção de resultados e discussões, é possível fazer uma estimativa do tempo de ciclo de scan de um programa, com as funções apresentadas, implementadas em separado. Logo, fica como sugestão para trabalhos futuros, estimar esse tempo de ciclo de *scan* com essas funções combinadas em um só programa. É interessante verificar também, a influência dos demais modos de temporizadores e contadores, e outras funções presentes na programação do CLP, no tempo do ciclo de varredura do mesmo.

Referências Bibliográficas

(Castrucci, 2007) MORAES, C. C.; CASTRUCCI, P. L. Engenharia de Automação Industrial. 2. ed. Rio de Janeiro, RJ: LTC, 2007

(Ovotron, 2018) CONTADOR DE OVOS COM RECONHECIMENTO POR IMAGEM. OVOTRON. Disponível em: <http://www.ovotron.com.br/produtos/>. Acesso em: 02 de jul. 2018.

(Parede, 2011) PAREDE, I. M.; GOMES, L. E. ELETRÔNICA: Automação Industrial. 1. ed. SÃO PAULO, SP: FUNDAÇÃO PADRE ANCHIETA, 2011

(Martins, 2007) MARTINS, G. M. Princípios de Automação Industrial. 2. ed. SANTA MARIA, RS: UNIVERSIDADE FEDERAL DE SANTA MARIA, 2007

(WEG, 2010) WEG. MANUAL DO USUÁRIO: MICRO CONTROLADOR PROGRAMÁVEL CLIC-02. São Paulo: 2010.

(Pedruzi, 2014) PEDRUZI, Gerson O. L., CONTROLE DE NÍVEL DE TANQUES CONECTADOS UTILIZANDO CLP CLIC-02 E ELIPSE SCADA. 2014. 58p. Trabalho de Conclusão de Curso – Universidade Federal de Viçosa, Viçosa, 2014.

(Brandão, 2012) BRANDÃO, A. S., et al. "Uma plataforma hardware-in-loop para vants de asas rotativas." *Proceedings of the XIX Congresso Brasileiro de Automática*. 2012.

(Gasparini, 2016) GASPARINI, Vitor Maggioni. Desenvolvimento de um modelo de controle de processo para altos-fornos a coque aplicado a altos-fornos industriais. 2016.

(CLIC02, 2018) Relés Programáveis CLIC02. WEG. Disponível em: <http://www.weg.net/catalog/weg/BR/pt/MKT_WDC_BRAZIL_PROGRAMMABLE_MICROCONTROLLER_PLC_CLIC02/>. Acesso em: 06 de jul. 2018.

(PLC300, 2017) WEG. MANUAL DO USUÁRIO: PLC300. São Paulo: 2017.

(TPW, 2017) WEG. MANUAL DO USUÁRIO: SÉRIE TPW-04. São Paulo: 2017.