PROCESSOR-IN-THE-LOOP APLICADO A MALHA DE VAZÃO DE UM SISTEMA CIP

M. S., COSTA[1], R. G. BRASILEIRO[2], E. S. GEDRAITE[3], R. GEDRAITE[4], T. S. OLIVEIRA[5]

[1,2,4,5] Universidade Federal de Uberlândia, Faculdade de Engenharia Química
[3] Siemens Brasil

RESUMO – *Neste artigo foi estudada e testada uma simulação do tipo Processor-in-the-Loop (PIL) no aplicativo Matlab/Simulnk™ usando uma placa Arduino UNO. Foi utilizado como modelo da planta um processo real de controle de vazão empregado na fase de enxágue em um processo Clean-In-Place tipicamente empregado na indústria de laticínios. A simulação do PIL foi feita por comunicação serial entre o aplicativo Matlab/Simulnk™ e o algoritmo do controlador de vazão implementado na placa Arduino, tendo sido verificado que esta conexão cria um atraso de resposta de cerca de 16 ± 6 segundos. No contexto das alterações trazidas pelos conceitos da indústria 4.0, este trabalho permitiu avaliar a comunicação serial entre dispositivos tipicamente empregados na área da automação industrial que empregam arquitetura aberta e que farão parte do cotidiano da indústria química a partir de agora. Apesar das dificuldades encontradas, foi possível estabelecer a comunicação serial com sucesso, permitindo evoluir no entendimento do tema e na futura aplicação dos conceitos de computação na nuvem.*

1. INTRODUÇÃO

À medida que a tecnologia evolui surgem novos produtos e processos mais robustos, complexos e dependentes de controladores automáticos. Algumas dessas evoluções requerem cada vez menos tempo entre suas atualizações, ou os testes de operação são extremamente difíceis e perigosos de serem efetuados. Portanto existe uma crescente necessidade de efetuar o maior número possível desses testes em ambientes virtuais.

Esses ambientes simulados já são utilizados há muito tempo, provavelmente um dos primeiros usos da técnica surgiu em meados de 1943 quando a companhia aérea *Pan Am* endossa a construção do primeiro simulador completo de uma aeronave, o Boeing 377, a ideia era estudar e testar a performance da aeronave sem a necessidade de decolar de verdade (PAGE, 2000). Um outro exemplo, mais atual, são as simulações realizadas antes de uma missão espacial, em que são testadas

as todas as condições de falha imagináveis como medida preventiva antes do seu lançamento (LEITNER, 1996), pois o custo desses testes seria consideravelmente superior caso fossem realizados literalmente.

A mesma técnica pode ser aplicada aos processos industriais como análise de condição de operação dos equipamentos ou projetos de modificações das linhas de produção, no qual os resultados podem servir como como análise de segurança e controle de processo, ou também, como apresentação para os acionistas e a gerência da empresa. Essa é uma tendência que ocorre nas empresas hoje, em que as indústrias buscam meios de se tornarem mais automatizadas e menos dependentes da ação humana a fim de diminuir gastos, riscos e erros, por isso, seus processos devem passar por simulações com todos os tipos de situações possíveis antes de sua operação.

À vista disto, observa a necessidade da aplicação da técnica também nos meios acadêmicos com a finalidade de melhor preparar os alunos para o mercado de trabalho atual. O estudo de simulações virtuais contribui para combinar e aperfeiçoar o entendimento de disciplinas como: Modelagem e Simulação de Processos, Controle de Processos, Segurança e Análise de Riscos, Operações Unitárias, Estatística, etc. Ajudando o aluno a desenvolver o senso crítico e analítico sobre um processo e proporciona uma visão aplicada do embasamento teórico apresentado por estas disciplinas.

O processo estudado neste trabalho é tipicamente usado em instalações industriais químicas, farmacêuticas e alimentícias e está relacionado com a limpeza de equipamentos de processo visando manter o processo isento de contaminações. Adicionalmente serão usados conceitos relacionados com a temática Indústria 4.0, buscando trazer este conteúdo para o dia a dia do estudante dos cursos de Engenharia Química.

O objetivo deste trabalho foi avaliar a comunicação entre o controlador do processo executado na plataforma Arduino e o modelo simulado no aplicativo Matlab/Simulink™ usando a metodologia denominada *Processor-In-the-Loop*, buscando identificar atrasos de comunicação.

2. *PROCESSOR-IN-THE-LOOP* (PIL)

A simulação PIL serve para testar a equivalência numérica entre o controle gerado em ambiente virtual (esperado) e o controle gravado em um sistema físico de processamento (MATHWORKS, 2020). Para esse tipo de teste, o código desenvolvido na linguagem C foi compilado para o hardware desejado e executado através de uma comunicação em loop com o modelo da planta em um ambiente simulado, no caso, o Simulink™, como pode ser observado na **Figura** 1. Os resultados obtidos podem ser utilizados para avaliar a performance do hardware comparativamente à da simulação SIL. Havendo diferença no desempenho, o código pode ser modificado considerando mudança do algoritmo de cálculo, liberação e alteração de variáveis e a troca de processador, entre outros.

O Simulink™ é um ambiente do aplicativo Matlab™ que provê um ambiente interativo e gráfico para modelagem e análise de sistemas dinâmicos. Os sistemas são criados em forma de blocos

de funções presentes na extensa biblioteca do aplicativo. O usuário simplesmente seleciona um dos blocos e o arrasta para uma tela, gerando uma comunicação entre eles. Ele permite a criação de complexos modelos com grande nível de detalhe feitos com pouco esforço do usuário, suporta modelos lineares e não lineares, contínuos e discreto.

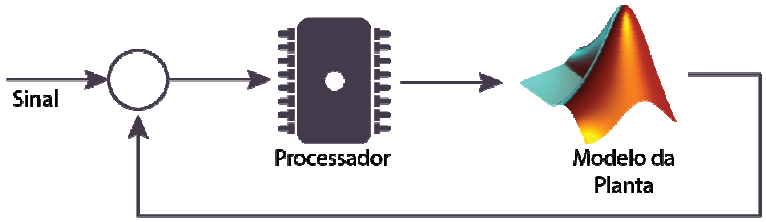

Figura 1 – Esquema de uma arquitetura PIL

O Simulink™ também permite a geração de código computacional, pois possui uma lista variada de compiladores que traduzem o sistema criado de forma gráfica em código C, C++, Phyton e outros. O aplicativo oferece suporte ao *Model Based Desing*, sendo capaz de testar a equivalência do código gerado, realizar a gravação e a comunicação em tempo real do programa em um hardware externo e seu ambiente virtual.

Foi empregada a plataforma de código aberto Arduino UNO para a função de PIL. Ela consiste em uma placa física, programável por meio do emprego de um software integrado, conhecido como IDE (*Integrated Development Environment*) baseado na linguagem C/C++ que permite ao Arduino se comunicar com outro dispositivo.

3. O MODELO DO PROCESSO

Foi utilizado como modelo do processo aquele desenvolvido por MELERO Jr. (2011) e CARNEIRO (2017) criado no ambiente simulink™ do aplicativo Matlab™ e apresentado na **Erro! Fonte de referência não encontrada.**, representada na cor laranja, a função de transferência da "Bomba 1" e um integrador de vazão para permitir a contabilização do gasto com água, expressa em Litros, durante a operação. O controlador de vazão com modos Proporcional e Integral também está inserido no diagrama de simulação do modelo.

O sistema considerado para implementação do teste do PIL foi o controle de vazão do sistema desenvolvido por MELERO Jr. et al. (2013). O processo simulado foi a função de transferência denominada "Bomba 1" e o controlador com modos Proporcional e Integral foi configurado na base de dados do Arduino UNO, usando a linguagem C. Adicionalmente, este mesmo sistema foi configurado no aplicativo Matlab/Simulink™, para fins de comparação de desempenho. Em ambos os casos foram utilizados os blocos de função correspondentes à comunicação serial disponíveis na

biblioteca de funções do Matlab/Simulink™. Na **Erro! Fonte de referência não encontrada.** é apresentado o diagrama de simulação correspondente, sendo a área assinalada no retângulo superior aquela correspondente ao aplicativo Matlab/Simulink™ e a área assinalada no retângulo inferior aquela correspondente à implementação do controlador na placa Arduino UNO. Importante destacar que as saídas fornecidas pelo diagrama de simulação da Figura 3 são a resposta do sistema a um degrau no sinal de entrada da planta simulada no aplicativo Matlab/Simulink™ e a diferença entre a resposta simulada no aplicativo Matlab/Simulink™ e a resposta gerada pelo Arduino UNO.

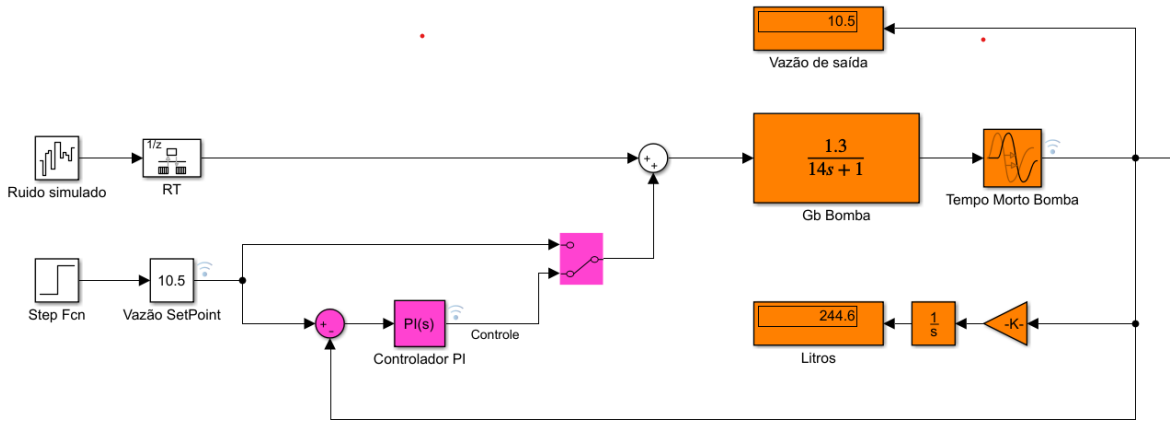

Figura2 – Diagrama de simulação do processo

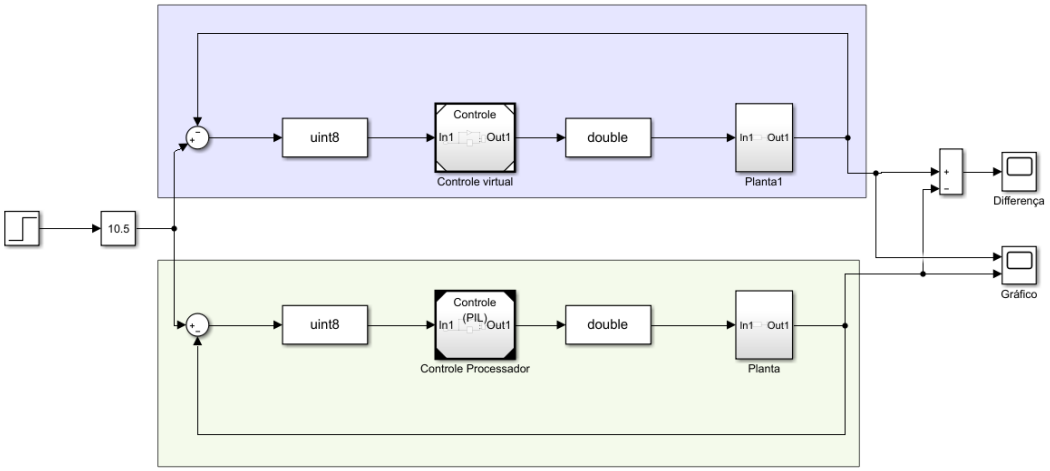

Figura 3 – Diagrama de simulação PIL com o aplicativo Matlab/Simulink™

4. METODOLOGIA EMPREGADA NOS TESTES

Após os testes preliminares com o sistema, feitos da maneira convencional com o emprego do ambiente Simulink™, foi decidido realizar o teste do PIL através da comunicação serial do Matlab™ com o Arduino. A porta USB da placa UNO permite a conexão serial UART (*Universal Asynchronous Receiver/Transmitter*) (FANG; CHEN, 2011), sendo que nesse tipo de comunicação os dados são enviados em série, um bit por vez. A cada 8 bits, portanto, é definido um byte de informação que é interpretado pelo receptor. Nesse tipo de comunicação é possível, também, estabelecer a velocidade de transmissão que os bits serão enviados e o receptor deve realizar a amostragem na mesma velocidade. Para a conexão serial entre o Arduino e o Simulink foi estabelecida uma velocidade de 9600 bits/s.

O teste consistiu em verificar a existência de um atraso inerente à comunicação entre a placa Arduino e o aplicativo Matlab/Simulink™, visando sua futura compensação. Como se pode observar na Figura , o sinal degrau é aplicado a dois ramais, sendo que em um deles (o superior) este sinal é processado integralmente no próprio Simulink™ e no outro (o inferior) é processado parcialmente no Simulink™ e parcialmente na placa Arduino, realizando as conversões necessárias antes de ser enviado à placa Arduino que o interpreta e devolve o sinal para o Simulink™ usando os blocos de função “**Saída Serial**” e “**Entrada Serial**” existente na biblioteca de funções do Simulink™ em conjunto com o bloco de função “**Config Serial**” também residente na referida biblioteca de funções. Os sinais processados em ambos os ramais são enviados ao bloco de função “**Mux**” e na sequência encaminhados para o bloco de função “**Scope**” visando a apresentação dos respectivos valores na forma gráfica.

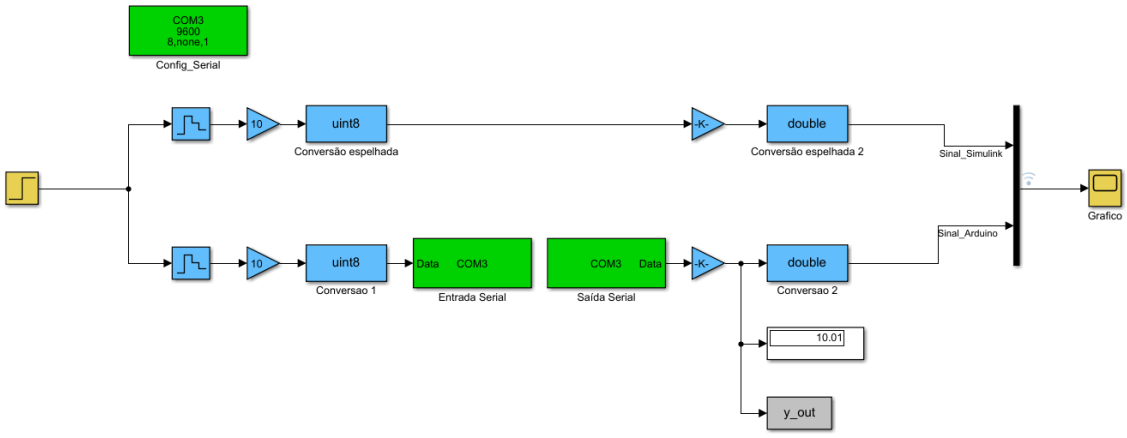

Figura 4 - Estabelecimento da Conexão Serial

5. RESULTADOS OBTIDOS

Os resultados dos testes realizados com a aplicação do degrau na entrada, no instante 10 segundos, foram coletados e analisados, permitindo verificar que existe um atraso de comunicação entre a placa Arduino e o aplicativo Matlab/Simulink™. Foi verificado, também, que este atraso varia a cada teste. No gráfico apresentado na **Erro! Fonte de referência não encontrada.** é mostrado que o sinal recebido da placa Arduino teve um atraso de aproximadamente (16 ± 6) segundos em relação ao sinal enviado pelo Simulink™ em cada teste. Observou-se também que no início de cada teste há um sinal espúrio, semelhante a um pulso, recebido pela placa Arduino, provavelmente gerado no estabelecimento da conexão com o Simulink™.

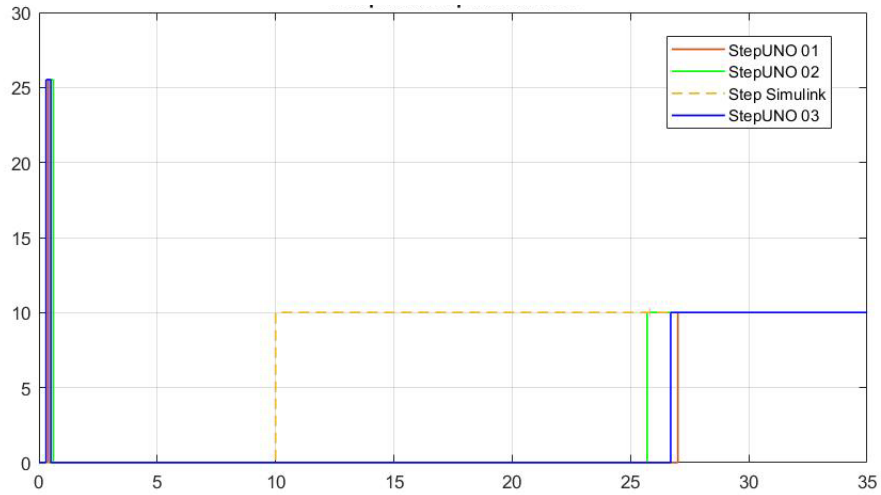

Figura 5 - Teste de comunicação para resposta a um degrau aplicado

Para se poder afirmar que existe um valor definido de atraso na comunicação serial estudada, foram realizados 50 testes, considerando as mesmas condições em todos eles. O resultado obtido é apresentado na Tabela 1.

Tabela 1 – Resposta a um sinal de entrada do tipo degrau aplicado em t = 10 s

Teste	t (s)	Teste	t (s)	Teste	t (s)	Teste	t (s)	Teste	t (s)
1	27,00	**11**	26,10	**21**	26,20	**31**	25,50	**41**	26,60
2	25,40	**12**	25,50	**22**	26,30	**32**	26,00	**42**	29,50
3	25,50	**13**	25,50	**23**	28,80	**33**	25,30	**43**	25,30
4	27,80	**14**	25,60	**24**	28,50	**34**	25,80	**44**	25,80
5	24,50	**15**	25,30	**25**	28,40	**35**	25,80	**45**	25,80
6	25,30	**16**	25,50	**26**	29,80	**36**	27,30	**46**	27,30
7	24,30	**17**	25,60	**27**	29,70	**37**	25,70	**47**	25,70
8	27,40	**18**	31,70	**28**	29,00	**38**	25,50	**48**	25,50
9	26,00	**19**	26,60	**29**	26,00	**39**	25,80	**49**	25,80

10	26,70	**20**	27,00	**30**	27,00	**40**	25,70	**50**	25,70

No entanto, não se pode afirmar, com segurança, que o tempo de resposta seja a média aritmética simples dos valores coletados e apresentados **Tabela 1** que seria no caso de 26,5080 segundos. Uma alternativa mais confiável seria o cálculo da distribuição normal dos valores coletados, a fim de determinar qual seria o tempo de resposta com maior probabilidade de ocorrência. Na **Figura** é apresentada a distribuição normal correspondente aos valores apresentados na Tabela 3, tendo sido encontrado o valor de 26,4639 segundos para o instante de resposta na comunicação.

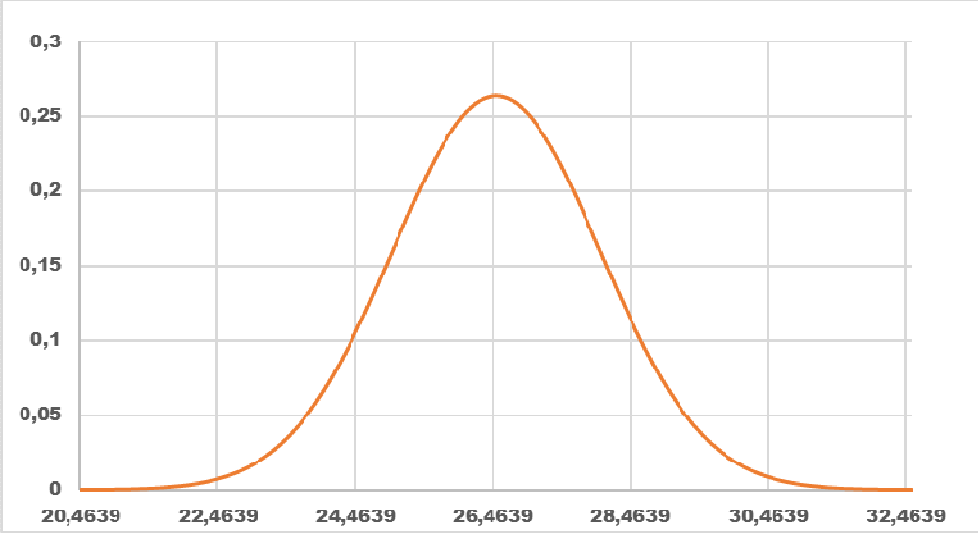

Figura 6 – Distribuição normal do atraso de comunicação

Com base nos resultados experimentais obtidos, foram propostos modelos matemáticos aproximados do tipo *FOPDT* (“First Order Plus Dead Time”) para representar o comportamento dinâmico das cinéticas de remoção de resíduos em função das vazões consideradas na etapa de enxágue, cujos parâmetros identificados são apresentados na sequência do presente trabalho.

6. CONCLUSÃO

Como observado durante a realização deste trabalho notou-se que a metodologia *Processor-in-the-Loop* apresentou limitações na troca de informações via comunicação serial entre a placa Arduino UNO e o Matlab/ Simulink™.

Para a viabilização dos testes foi necessário a utilização de comunicação serial de menor precisão devido ao fato do bloco de comunicação serial de “double precision” não ter funcionado satisfatoriamente no Simulink™. Optou-se por utilizar o bloco “uint8” para a função de comunicação.

Foi possível quantificar o valor do atraso na comunicação serial entre o Matlab/ Simulink™ e a placa Arduino UNO de maneira satisfatória, com boa precisão, tendo sido verificado que o valor encontrado para o atraso de comunicação não representa – em princípio – um problema significativo para o correto funcionamento do sistema testado.

Como recomendação de trabalhos futuros, sugere-se expandir os estudos iniciados neste trabalho, considerando o emprego de um hardware com maior capacidade de memória do que o Arduino UNO, que permita a realização do teste *Processor-In-the-Loop* na sua forma plena, considerando o emprego do aplicativo Matlab/Simulink™.

Sugere-se, também, a melhoria do sistema estudado, considerando o emprego da metodologia *Hardware-In-the-Loop*, sendo assim possível melhorar o desempenho do sistema de controle do processo *Clean-in-Place*.

6. REFERÊNCIAS

CARNEIRO, L. R. A. **Ajuste dinâmico dos parâmetros do controlador de vazão de água de enxágue aplicado à remoção da solução detergente em sistema CIP**. Universidade Federal de Uberlândia, Uberlândia, 2017.

FANG, Y. Y.; CHEN, X. J. **Design and simulation of UART serial communication module based on VHDL**. In: *Proc. of 3rd Int. Work. Intell. Syst. Appl.* ISA *2011*, vol. 1, 2011.

LEITNER, J. **Space technology transition using hardware in the loop simulation**. *IEEE Aerosp. Appl. Conf. Proc.*, vol. 2, pp. 303–311, 1996.

MATHWORKS. **Code Verification and Validation with PIL and External Mode**. 2020. Available: https://www.mathworks.com/help/supportpkg/beaglebone/examples/code-verification-and-validation-with-pil-and-external-mode.html. [Accessed: 17-Oct-2020].

MELERO Jr., V. , GEDRAITE, E. dos S., KUNIGK, L, VIEIRA, P. A., MALAGONI, R. A., SISLIAN, R., COUTINHO Filho, U., GEDRAITE, R**.** e AUGUSTO, S. R. **Experimental investigation about rinse water consumption of a CIP process applied to a shell and tube exchanger.** In: *Proc. of 3rd International Conference on Chemical Engineering and Advanced Materials (ICCEAM – 2013)*. July 6-7, 2013, Guangzhou, China.

MELERO Jr., V. **Instrumentação e identificação de um processo de sanitização cinética CIP**. Escola de Engenharia Mauá, São Caetano do Sul, 2011.

PAGE, R. L. **Brief history of flight simulation**. *SimTecT 2000 Proc.*, pp. 1–11, 2000.