Sistema de irrigação automatizado auxiliado por sistema computacional de tempo real baseado em Arduino

Lucas F. Vieira[1]

[1]Faculdade de Tecnologia e Ciências Socias Aplicadas – Centro Universitário de Brasília (UniCEUB)
Brasília – DF – Brazil

llucasfernandes@live.com

Abstract. *This paper presents a prototype of a real-time automated irrigation system based on the Arduino platform that enables the delivery of a defined and scheduled amount of water for two way channels. The prototype delivers the expected volume by an error margin of 12% in the most discrepant case after several measurements with a marked container and is a considerable alternative for domestic irrigation systems.*

Resumo. *Este trabalho apresenta um protótipo de um sistema de irrigação automatizado de tempo real baseado na plataforma Arduino, que permite o fornecimento de um volume de água definido e de maneira agendada para dois canais. O protótipo entrega o volume esperado com uma margem de erro de 12% no caso mais discrepante após seguidas medições com um recipiente marcado e é uma alternativa considerável para sistemas de irrigação domésticos.*

1. Introdução

Sistemas automatizados de irrigação são alvo de estudos há bastante tempo e são uma entre as diversas aplicações de sistemas de tempo real em ambientes físicos. Essa categoria de sistema é largamente utilizada em atividades de controle por ter como característica fundamental a previsibilidade, em dois aspectos críticos. O primeiro é a conformidade com os regimes de tempo estabelecidos durante o projeto e o segundo é a tolerância a falhas, com tratamentos também estabelecidos previamente. Tais características fazem dos sistemas de tempo real apropriados para sistemas críticos e de controle, e também são a razão de sua escolha para este trabalho.

Este trabalho detalha a construção do protótipo de um sistema de irrigação controlado por sistema de tempo real baseado na plataforma Arduino, com foco na precisão do volume de água, bem como no respeito ao agendamento deste fornecimento, para fins acadêmicos. Devika et al [1] implementam um sistema bastante similar em seu artigo, porém, com foco no mantimento de um estado esperado de umidade para o solo monitorado por um sensor específico. Em contraste, o protótipo apresentado neste trabalho não utiliza nenhum sensor, e as atividades de controle são possíveis através da marcação temporal, nativa ao próprio equipamento escolhido, em

conjunto com medições sobre os equipamentos controlados durante o projeto do protótipo.

2. Referencial teórico

Este trabalho aborda termos e conceitos específicos pertinentes ao universo dos sistemas de tempo real e alguns equipamentos utilizados no protótipo, este capítulo aborda suas definições.

2.1. Sistemas de tempo real (STR)

“Um Sistema de Tempo Real (STR) é um sistema computacional que deve reagir a estímulos oriundos do seu ambiente em prazos específicos.” [2]

2.2. Tolerância a falhas

“Tolerância a falhas se destina a preservar a prestação do serviço correto na presença de falhas ativas. Isto geralmente é implementado através da detecção de erros e subsequente recuperação do sistema” [3]

2.3. Tarefa

“O conceito de tarefa é uma das abstrações básicas que fazem parte do que chamamos um problema de escalonamento. Tarefas ou processos formam as unidades de processamento seqüencial que concorrem sobre um ou mais recursos computacionais de um sistema. Uma simples aplicação de tempo real é constituída tipicamente de várias tarefas. Uma tarefa de tempo real, além da correção lógica ("correctness"), deve satisfazer seus prazos e restrições temporais ou seja, apresentar também uma correção temporal ("timeliness").” [4]

2.4. Período

Diferença regular de tempo entre duas ocorrências de uma tarefa periódica.

2.5. Deadline

“Deadline é um marco de tempo.” [5].

2.6. Tempo de computação

Tempo de processamento necessário para executar uma tarefa até o seu fim.

2.7. Região crítica

Espaço de tempo em que uma tarefa acessa um recurso limitado do sistema. Para este protótipo, o fornecimento de água é um recurso limitado a apenas um acesso por vez, então, quando uma tarefa acessa esse recurso ela está acessando uma região crítica do sistema.

2.8. Restrições de tempo

Limitações temporais às quais um sistema de tempo real deve respeitar. A rigidez com que são tratadas as falhas no cumprimento das restrições temporais classifica-os em Soft

Real-Time, em que os dados não são considerados totalmente inválidos após a perda de um ou mais deadlines; Firm Real-Time , em que os dados podem se tornar inválidos após a perda de alguns deadlines; e Hard Real-Time, onde nenhum deadline pode ser perdido, ou seja, o estado do sistema é inválido e as informações inúteis após a perda de qualquer deadline. O protótipo deste trabalho é Soft Real-Time.

2.9. Paradigmas de programação para sistemas de tempo real

Por uma limitação imposta pela plataforma Arduino, a codificação deve ser, obrigatoriamente, na linguagem C (paradigma procedural). Em geral, qualquer paradigma de programação pode ser utilizado em sistemas de tempo real, desde que esteja em conformidade com as limitações desse tipo de sistema. Várias linguagens possuem módulos ou bibliotecas que especializam o desenvolvimento para sistemas de tempo real, como o JRockit, para Java. Dessa forma, a escolha da mais adequada varia de acordo com as necessidades e limitações do sistema desejado.

2.10. Arduino

"(...) um Arduino é um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele." [6]

3. Desenvolvimento

Algumas observações devem ser feitas antes de detalhar a implementação pois justificam as escolhas feitas.

1. O fornecimento de água para o sistema é considerado infinito.
2. Apesar de infinito, o fornecimento de água para o sistema só é suficiente para um dos canais de irrigação acessar, concomitantemente.
3. A duração da ativação do fornecimento para cada canal de irrigação possui um tempo limite, de modo que os canais recebam água alternadamente.
4. Se um dos canais atingir o volume de água esperado para o período, ele deixa de disputar o fornecimento até a nova janela de ativação.
5. O volume de água necessário e a frequência de fornecimento não são baseados em um tipo específico de vegetação, os valores podem ser configurados conforme desejado, desde que sejam dados computáveis de acordo com o algoritmo de escalonamento de Taxa Monotônica (RM).
6. O fornecimento de energia elétrica para o sistema é considerado ininterrupto.

Existem quatro componentes ativos neste protótipo: um módulo Arduino UNO, que realiza o controle computacional do conjunto; um módulo com dois relés próprio para uso em conjunto com módulos Arduino, que faz o chaveamento da alimentação das bombas de água, ligando e desligando-as; e duas bombas de água submersíveis, comumente utilizadas em aquários de pequeno porte, que fornecem o volume de água esperado ao longo de dois canais, sendo que cada canal está conectado a uma bomba. O protótipo foi construido conforme o diagrama da Figura 1 demonstra.

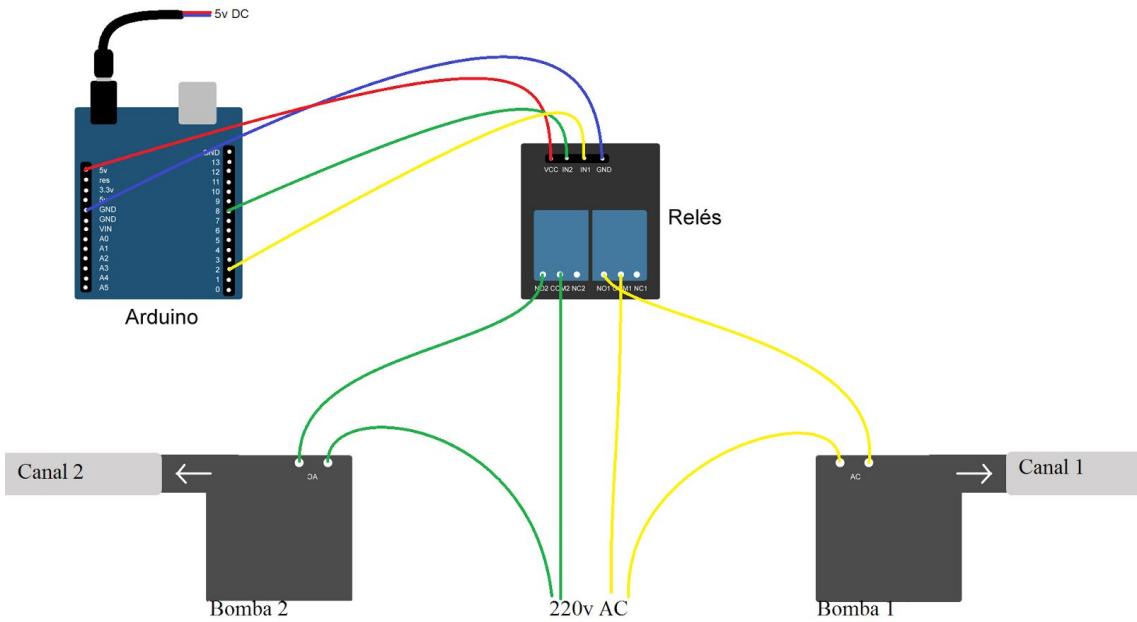


Figura 1. Diagrama

Em virtude da não utilização de sensores no protótipo, o controle do volume de água pelas bombas é feito baseado no tempo de acionamento das mesmas. Foram realizadas cinco medições para cada uma das bombas sobre o tempo necessário para fornecer 0,5L de água, em um recipiente marcado, estando a bomba em pleno funcionamento. Em seguida, o mesmo teste foi feito, porém com as bombas, de início, desligadas, como forma de marcar o seu tempo de resposta pela diferença entre este tempo e o tempo obtido anteriormente. Os dados obtidos encontram-se na Tabela 1 e na Tabela 2, respectivamente..

Tabela 1. Tempo, em segundos, necessário para fornecer 0,5L.

Amostra	Bomba 1	Bomba 2
Amostra 1	17,99	54,18
Amostra 2	18,67	53,62
Amostra 3	18,28	53,84
Amostra 4	18,77	54,55
Amostra 5	19,27	53,57

A partir destes dados, o volume fornecido por segundo foi calculado pela média aritmética das cinco amostras. Para a bomba 1 representa o valor de 0,0268L/s, enquanto para a Bomba 2 representa o valor de 0,0092L/s. No código, o valor foi convertido para mL/s, sendo 26,8mL/s e 9,2mL/s, respectivamente.

Tabela 2. Diferença de tempo entre a amostra obtida pela bomba em pleno funcionamento e a amostra obrida com a bomba, a princípio, desligada.

Amostra	Bomba 1	Bomba 2
Amostra 1	0,95	0,99
Amostra 2	0,98	0,96
Amostra 3	0,93	0,94
Amostra 4	0,9	0,95
Amostra 5	0,99	0,91

Já a partir destes valores, foram obtidos o tempo médio de resposta pela média aritmética das cinco amostras. Em ambas as bombas, isto representou o valor de 0,95s.

A partir destes dados, foram definidas as duas tarefas que o sistema possui, sendo uma para cada bomba/canal, descritas na Tabela 3. As tarefas serão escalonadas segundo o algoritmo de Taxa Monotônica (RM), que é o utilizado pelo Nil RTOS, logo, o período coincide com o deadline. A tabela inclui, ainda, o tempo que deve ser acrescido ao tempo de computação em virtude do tempo de resposta do acionamento das bombas. Este tempo é calculado obtendo o número de ativações necessárias até o fim da tarefa, por conta do limite de 3 segundos de ativação contínua, multiplicado pelo tempo de ativação das bombas. Para a bomba 1, por exemplo, são necessárias 4 ativações com tempo limite de 3 segundos cada para fornecer o volume esperado. Portanto, há um acréscimo de 4 x 0,95 = 3,8 segundos ao tempo de computação.

Tabela 3. Tarefas do sistema

Bomba/Canal	Tempo de computação	Período	Deadline
1	9,32s (250mL) + 3,8 = 13,12s	60s	60s
2	10,86s (100mL) + 3,8 = 14,66s	60s	60s

As tarefas definidas são escalonáveis segundo o teste de escalonabilidade do RM, na fórmula da Figura 2.

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} \leq n(2^{1/n} - 1)$$

Figura 2. Fórmula do teste de escalonabilidade do algoritmo RM

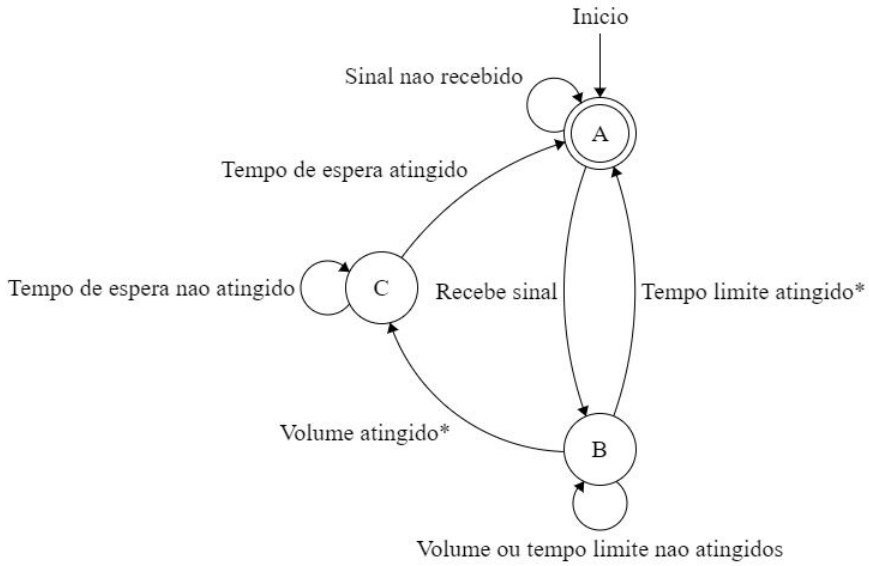
As tarefas são escalonáveis se (13,12 / 60) + (14,66 / 60) <= 2(2^(1 / 2) - 1):

(13,12 / 60) + (14,66 / 60) = 0,218 + 0,244 = 0,462 e

2(2^(1 / 2) - 1) = 2(1,41 - 1) = 0,828.

Como 0,462 <= 0,828, o conjunto é escalonável pelo algoritmo de Taxa Monotônica.

O algoritmo para o sistema de controle foi feito baseado no grafo do autômato da Figura 3.



* - Sinal enviado
A - Bomba de água desligada, espera por sinal de acesso.
B - Bomba de água ligada, espera por tempo limite ou volume necessário serem atingidos.
C - Bomba de água desligada, espera pela nova janela de ativação no período seguinte.

Figura 3. Autômato

A programação do módulo Arduino para o controle do protótipo foi feita na linguagem C, nativa do equipamento, em conjunto com a biblioteca Nil RTOS, que implementa as funções básicas de um sistema operacional de tempo real. O sistema base foi feito por Giovanni Di Sirio, e o porte para Arduino por Bill Greiman. O código fonte utilizado no protótipo encontra-se disponível em http://pastebin.com/1QBAdX2j.

4. Resultados

Foram realizados 5 testes em cada uma das bombas, individualmente, em um recipiente com capacidades marcadas e anotadas as diferenças entre o valor esperado e o valor obtido, sendo que o valor esperado é o volume definido para o período, de 500mL para a bomba 1 e de 100mL para a bomba 2. Os dados obtidos seguem na Tabela 4.

Tabela 4. Resultados obtidos

Amostra	Bomba 1 (500mL)	Bomba 2 (100mL)
Amostra 1	+15mL	+10mL
Amostra 2	-25mL	+25mL

Amostra 3	**-50mL**	**0mL**
Amostra 4	**0mL**	**0mL**
Amostra 5	**+50mL**	**+25mL**

A partir destes dados é possível definir um valor esperado de erro de -2mL, para a bomba 1 e de 12mL, para a bomba 2, tendo as médias aritméticas dos valores. A margem de erro esperada no pior cenário após seguidas medições é de 12%.

5. Conclusão

O protótipo demonstra ser uma alternativa viável para sistemas de irrigação domésticos, tendo como principal vantagem o agendamento e a flexibilidade para atender a diferentes demandas de fornecimento. O volume esperado é satisfeito por uma margem compatível com esta aplicação. O protótipo também pode ser aproveitado em melhorias futuras para suportar mais canais de fornecimento, maior precisão de volume e implementado, de fato, em um ambiente para uso prático.

6. Referências

[1] S.V. Devika, S. Khamuruddeen, S. Khamurunnisa J. Thota and K. Shaik (2014) “Arduino Based Automatic Plant Watering System”, International Journal of Advanced Research in Computer Science and Software Engineering

[2] [4] J. M. Farines,J. S. Fraga and R. S. Oliveira (2000) “Sistemas de Tempo Real”, páginas 4 e 12, respectivamente.

[3] A. Avizˇienis (2012) “Fundamental Concepts of Dependability”, página 7.

[5] A. P. Magalhães, M. Z. Rela, J. G. Silva (1993) “Deadlines in Real-Time Systems”, página 2.

[6] M. McRoberts (2011) “Arduíno Básico”, página 22.