Filipe Travizani Ricato
Pedro Henrique Bonifácio

Desenvolvimento de Código Não Bloqueante para Sistemas Embarcados

Projeto de Linguagem de Programação

Matéria: Linguagem de Programação
Professor: Fernando Simplício

São Paulo
2018

1 Objetivos

Este projeto foi desenvolvido para exemplificar os conceitos apresentados na matéria **Linguagem de Programação**. No decorrer das aulas ministradas, foram apresentados desde conceitos básicos da linguagem de programação C como funciona o processo de compilação, os diversos tipos de dados, laços de repetição, estruturas básicas de um programa e suas bibliotecas, matrizes, vetores e funções.

Também foram abordadas metodologias de como tornar um sistema mais responsivo a eventos e eficiente no processamento de suas funções. Dentre estas metodologias foram apresentadas formas de separar funções em diversos blocos de processamento e também o uso de um sistema operacional de tempo real. Portanto, neste trabalho buscou-se reformular as bibliotecas de periféricos fornecidas pelo professor, utilizando máquinas de estados, de modo que as funções utilizadas no *loop* principal sejam executadas de forma contínua e não bloqueante.

2 Justificativa

Sistemas embarcados podem ser facilmente encontrados no dia a dia das pessoas, tornando-se parte do cotidiano. Com a evolução de setores como indústria 4.0 e internet das coisas, o desenvolvimento de dispositivos mais eficientes, inteligentes e poderosos torna-se cada vez mais necessário. O componente capaz de tornar um simples microprocessador em uma máquina inteligente é o *firmware* programado neste, portanto a qualidade e confiabilidade de um produto está intimamente relacionada ao código desenvolvido para este.

O uso de algoritmos não bloqueantes em um *software* busca evitar que tarefas não sejam executadas no momento correto devido ao programa estar travado em apenas uma tarefa. Ao empregar este método de codificação busca-se evitar que o processamento de informações ou a espera por um recurso cause erros ou falhas em outros processos ou *threads*. A utilização de algoritmos deste tipo se faz extremamente necessárias em sistemas de tempo real restritivos (*hard real time systems*) onde todas as tarefas devem ser executadas dentro de seus limites de tempo.

3 Metodologia

A metodologia do trabalho consiste em algumas etapas: primeiro deve-se realizar uma pesquisa sobre métodos para tornar o sistema mais responsivo a eventos e efetivo no processamento de várias funções. Após a pesquisa é determinado qual método será utilizado, neste caso, foram utilizadas funções não bloqueantes baseadas em *buffers* circulares, máquinas de estados finitos, para realização das tarefas. O segundo passo consiste em reescrever as funções fornecidas pelo professor de tal maneira a adequar a proposta de ter um código não bloqueante. Por fim para validação do código elaborado foi utilizado a placa do arduino *ATMEGA2560*, e *Display LCD Shield* com teclado.

4 Desenvolvimento

Para o desenvolvimento deste projeto foram abordadas algumas técnicas e estruturas características de programas não bloqueantes, como a implementação de máquinas de estados, *buffers* circulares, divisão de funções em vários processos à partir do conceito de atomicidade, ou processos indivisíveis, além de empregar variáveis estáticas para funções para armazenar o estado de processamento atual.

Buffer Circular

Para facilitar a troca de dados, torna-se interessante utilizar um buffer FIFO (*first in, first out*) para que os dados sejam processados na ordem que foram recebidos. Para implementar um *buffer* circular são necessários duas variáveis indexadoras, indicando o inicio e fim dos dados, e um *buffer* de dados. Abaixo, segue um exemplo da definição de um *buffer* circular em linguagem C.

```
typedef struct
{
    uint8_t data[100];
    uint16_t p_next_empty;
    uint16_t p_next_send;
} circular_buffer_t;
```

Máquinas de Estado

O uso de máquinas de estados finitos é essencial para executar um programa de forma sequencial não-bloqueante. Normalmente, cada estado representa a execução de uma tarefa, ou a espera de um processo de um periférico ser finalizado. Normalmente implementa-se diferentes máquinas de estado para cada tarefa ou periférico do sistema. Abaixo tem-se um exemplo da implementação de uma máquina de estados.

```
void USART_SM(void)
{
    static usart_sm_status_t status_sm_rx = READ;
    switch(status_sm_rx)
    {
        case READ:
            ...
        break;
        case CRC_CALC:
            ...
        break;
        case HANDLE:
             ...
        break;
    }
}
```

5 Conclusões

Ao trabalhar com técnicas de codificação não bloqueante, é gerado um aumento na confiabilidade do funcionamento do código, diminui-se a chance de possíveis perdas de dados ou falhas em execução de processos. O desenvolvimento de soluções deste tipo pode ser mais demorado devido à complexidade empregada nos algoritmos e sequências lógicas; porém, ao utilizar práticas consolidadas este tempo tende à diminuir consideravelmente.

Referências

[1] KLEIN, Derek Caleb. Non-Blocking Hardware Coding for Embedded Systems. 2011.