UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA
DEPARTAMENTO DE TECNOLOGIA
ÁREA DE ELETRÔNICA

Problema #1 – Linguagem assembly

1. Tema

Produção de código executável a partir de códigos fonte e objeto feitos em Assembly.

2. Objetivos de Aprendizagem

Ao final da realização deste problema, o/a discente deverá ser capaz de:

- Programar em Assembly para um processador com arquitetura ARM;
- Entender o conjunto de instruções da arquitetura ARM e saber como utilizá-las de acordo com a necessidade do sistema;
- Entender como integrar código assembly e códigos C para produzir um programa executável;
- Avaliar o desempenho de um código assembly através de medidas sobre o comportamento de sua execução no sistema.

3. Contexto

Embora incomum, a linguagem assembly é a única linguagem de programação que permite controle absoluto sobre o hardware. Ainda que não seja tão amigável quanto linguagens como JAVA, JavaScript, Python ou PHP, essa linguagem existe desde os primórdios dos sistemas computacionais e dificilmente deixará de existir no futuro, já que é através dela que se consegue acessar todos os recursos do hardware.

A construção de qualquer sistema computacional passa pela programação de rotinas de controle explícito do hardware, que são feitas sempre em linguagem assembly. Além disso, softwares de base, como o sistema operacional e o compilador, ou softwares de infraestrutura, como servidores Web, sistemas de gerenciamento de banco de dados, e motores de jogos, requerem que os programadores desenvolvem trechos de código capazes de extrair o máximo de desempenho possível do hardware, o que muitas vezes só é possível através da linguagem assembly.

Não bastasse isso, qualquer hacker ou profissional voltado a segurança de softwares e de sistemas precisará obrigatoriamente entender de linguagem assembly para realizar suas atividades, seja "hackeando" um sistema ou implementando formas que evitem que o sistema seja 'hackeado".

Conhecer a linguagem assembly de um sistema computacional é a melhor maneira de aprender sua organização básica. Os programadores que se esforçam para dominar a linguagem assembly tornam-se melhores programadores de linguagem de alto nível. Sua capacidade de escolher implementações de alto nível apropriadas para produzir código eficiente, sua capacidade de ler código de linguagem de alto nível e detectar bugs hediondos

em um sistema, e sua compreensão de como todo o sistema opera os eleva a um *status* quase lendário entre seus pares.

A interface RS-232 (também conhecido por EIA RS-232C ou V.24) é um padrão de protocolo para troca serial de dados binários entre dois equipamentos: um DTE (*Data Terminal Equipment*) e um DCE (*Data Communication Equipment*). Este padrão foi originalmente usado nos primórdios dos sistemas computacionais para conectar um teletipo (equipamento eletromecânico de comunicação assíncrona que usava código ASCII) a um modem (que fazia a transmissão de dados). O sistema foi evoluindo, e a terceira revisão deste padrão (chamada de RS-232C) foi publicada em 1969, para adequar às características elétricas dos diversos dispositivos da época. Posteriormente PCs (e outros equipamentos) começaram a utilizar este padrão para comunicação com equipamentos já existentes. Quando a IBM lançou computadores com uma porta RS-232, esta interface foi popularizada, e perdurou por muitos anos como padrão para comunicação serial em quase todos os computadores. Atualmente, os PCs utilizam o padrão USB para comunicação com seus periféricos.. Porém, o RS-232 ainda é muito utilizado na indústria para a comunicação simples e confiável entre dispositivos, como por exemplo microcontroladores. (Baseado no texto “RS-232”, disponível em: https://pt.wikipedia.org/wiki/RS-232)

4. Problema

“Mercado de IoT movimentará mais de 30 bilhões de dólares na América Latina até 2023”[1] Visando uma fatia desse mercado, você e sua equipe foram contratados para implementar o protótipo de um sistema digital baseado em um processador ARM, que recebe informações de sensores através de sua UART. O protótipo deve ser o mais simples e energeticamente eficiente possível, sem abrir mão da funcionalidade e elegância nas soluções propostas. Para isso, testes iniciais deverão ser realizados utilizando o Raspberry Pi Zero, programado em linguagem assembly.

5. Requisitos

O sistema de comunicação a ser desenvolvido na Raspberry Pi Zero deve atender às seguintes restrições:

- 5.1. O código deve ser escrito em Assembly. Posteriormente, esse código será usado em conjunto com o restante do sistema que será desenvolvido em C;
- 5.2. O sistema deve permitir as seguintes configurações:
  - 5.2.1. Velocidade (*baud rate*);
  - 5.2.2. Paridade;
  - 5.2.3. Quantidade de bits de parada (*stop bits*);
  - 5.2.4. Quantidade de bits da mensagem.
- 5.3. Deve ser realizado teste de *loopback* para validação do código.

6. Produto

No prazo indicado no cronograma a seguir, cada equipe deverá apresentar:

- 6.1. Código
  - 6.1.1. Código em linguagem Assembly;

[1] https://tiinside.com.br/26/02/2020/mercado-de-iot-deve-ultrapassar-us-30-bilhoes-na-america-latina-ate-2023/

6.1.2. Todos os códigos deverão estar detalhadamente comentados;
6.2. Script de compilação tipo Makefile para geração do código executável;
6.3. Relatório técnico contendo, no mínimo:
6.3.1. Software usados, incluindo softwares básicos;
6.3.2. Arquitetura do computador usado nos testes;
6.3.3. Descrição dos tipos de instruções utilizadas;
6.3.4. Descrição dos testes de funcionamento do sistema, bem como, análise dos resultados alcançados.

7. Avaliação

Para avaliar o envolvimento do grupo nas discussões e na apresentação, o tutor poderá fazer perguntas variadas a qualquer membro, tanto nas sessões tutoriais quanto na apresentação.

A nota final será a composição de 3 (três) notas parciais:

Critério	Critérios para a nota	Peso
Desempenho Individual	Participação individual nas sessões tutoriais, de acordo com o interesse e entendimento demonstrados pelo aluno, assim como sua assiduidade, pontualidade e contribuição nas discussões.	4
Documentação	Relatório técnico de cada grupo, considerando qualidade da redação (ortografia e gramática), organização dos tópicos, definição do problema, descrição da solução, explicação dos experimentos, análise dos resultados e conclusões.	3
Códigos	Qualidade do código fonte (organização e comentários), e execução correta dos códigos binários de acordo com testes de validação que explorem as situações de uso.	3

8. Cronograma

Semana	Data	Descrição
1	qua,09/03/22	Problema 1 – Apresentação
	sex,11/03/22	Problema 1 – Seção Desenvolvimento #1
2	qua,16/03/22	Problema 1 – Seção Tutorial #2
	sex,18/03/22	Problema 1 – Seção Desenvolvimento #2
3	qua,23/03/22	Problema 1 – Seção Tutorial #3
	sex,25/03/22	Problema 1 – Seção Desenvolvimento #3

4	qua,30/03/22	Problema 1 – Seção Tutorial #4
	sex,01/04/22	Problema 1 – Seção Desenvolvimento #4
5	qua,06/04/22	Problema 2 – Apresentação
	sex,08/04/22	Problema 1 – Entrega/Avaliação