**Nomes:** Ederson dos Santos Fernandes

Diego Wesley Braga

**Disciplina:** Arquitetura e Organização de Computadores.

**Estudo do Assembly do Processador ColdFire**

* **Estrutura e Nome dos Registradores**

O processador ColdFire contém 16 registradores de 16 bits, de forma que são subdivididos em Registradores de Dados (Data Registers), constituídos por 8 registradores nomeados por D7 – D0, respectivamente; e Registradores de Endereço(Address Registers), constituídos por 8 registradores nomeados por A7 – A0, respectivamente.

Os registradores de dados são utilizados para operações com bit, byte(8 bits), word(16 bits) e long-word (32 bits). Eles também podem ser usados ​​como registradores de índice(index registers).

Os registradores de endereço são utilizados como apontadores de pilha de software, registradores de índice ou registradores de endereço de base. Os registos de endereços de base podem ser utilizados para operações de word e long-word. o registrador A7 funciona como um ponteiro de pilha de hardware durante o empilhamento para chamadas de sub-rotina.

* **Instruções Originais do Processador**
  + **Carga de constante: MOVEQ - Move Quick.**

**Operação:** Dado Imediato **→** Destino

**Assembler Syntax:** MOVEQ # < data > ,Dn

Sendo: -Dn: Qualquer registrador de dados n.

* + **Cópia de Valor Entre Registradores:MOVE/MOVEA - Move dados da origem para o destino**

**Operação:** Origem **→** Destino

**Assembler Syntax:** MOVE <ea> , <ea> ou MOVEA <ea>, An

Sendo: - An: Qualquer registrador de endereço n.

- ea: Um endereço efetivo

* + **Soma de dois valores: ADD - Adição**
  + **MUDAR PARA ADDX!!!!!!!**

**Operação:** Origem + Destino → Destino

**Assembler Syntax:** ADD <ea> ,Dn ADD Dn, <ea>

Sendo: - Dn: Qualquer registrador de dados n.

- ea: Um endereço efetivo

* + **Subtração de dois valores: SUB - Subtração**
  + **MUDAR PARA SUBX!!!!!!!**

**Operação:** Destino - Origem → Destino

**Assembler Syntax:** SUB <ea> ,Dn SUB Dn, <ea>

Sendo: - Dn: Qualquer registrador de dados n.

- ea: Um endereço efetivo

* + **Desvio incondicional: JUMP - Salto incondicional**

**Operação:** Endereço de Destino → PC

**Assembler Syntax:** JMP <ea>

Sendo: - ea: Um endereço efetivo