Lista de Exercícios

Aluno: Leonardo Luiz Freitas de Resende M: 15.1.5773

1. Aparelhos como Smartphones, notebooks, servidores até projetos embarcados de pequeno porte, entre outros.
2. É um conjunto de fácil implementação comparado a outras alternativas e o projeto possui grande aceitação na indústria de semicondutores. Tem como objetivo um conjunto de instruções 'universal' que é livre e aberto para todos os usuários, provendo tudo que é necessário para suportar perfeitamente qualquer projeto comercial, incluindo fazer design, fabricar e vender os chips e software RISC-V e que rode programas 128 bits e inferiores.
3. Não, diferentemente das outras opções no mercado, o RISC-V pode ser utilizado para experimentação e uso acadêmico.
4. Multiplicação M:
   - ➔ Adiciona 4 instruções de multiplicação, duas de divisão, e duas de manipulação de restos.
   - ➔ A base é indicada pela base do ISA.
   - ➔ A divisão por 0, por exemplo, não causa interrupção, mantendo a simplicidade.

Sincronização A:
Adiciona 11 instruções de sincronização visando consistência e atomicidade da operação. São divididos em dois grupos: Load Reserved e Store Conditional para operações atômicas na memória.

Extensão: Ponto Flutuante

Possui três extensões diferentes: Precisão simples (F), Precisão Dupla (D) e quádrupla precisão (Q). A extensão F é pré-requisito para D, que é pré-requisito para Q.

Introduz também 32 registradores de ponto flutuante (f0-f31) de tamanho 32 bits e um registrador de 5 bits para exceções. Exceções não geram interrupções e com isso devem ser verificadas por meio de consultas.

As instruções load–store usam o mesmo endereçamento base+offset.

Extensão: Compressão De Tamanho De Código

A última extensão C não adiciona nenhuma outra função, mas, ao invés disso, codifica as instruções inteiras para salvar espaço e com isso reduzir o tamanho do footprint. É disponível para bases inteiras, bem como load e store para pontos flutuantes e cria-se então instruções comprimidas em 16 bit. Basicamente, cada função compactada é mapeada diretamente à instrução real e possui algumas restrições para a compressão sobre o formato dos operandos.

O propósito de reduzir o tamanho do código binário, energia e custo para pequenos computadores, é visando sistemas embarcados. Pesquisas com C mostram que um código 20\% menor que um x86 e MIPS Comprimido e 2% maior que um ARM Thumb-2.

5. Sim, pois contém uma arquitetura que atende a todos os projetos de hardware. Para ativar as extensões, porções do espaço de codificação de instruções já foram reservados para uso futuro.
6. Possui 32 registradores inteiros, e 32 registradores opcionais para ponto flutuante. Também existe uma variante RISC-V com 16 registradores inteiro e sua memória possui endereçamento de 1 byte.
7. 1) Colocar os bits mais significantes numa posição fixa;
   2) Disposição de bits para reduzir o número de multiplexadores no CPU;
8. RISC-V, intencionalmente, não possui códigos condicionais, nem mesmo bit de carry e sendo assim, os projetistas afirmam que isso pode simplificar o desenvolvimento do CPU, minimizando interações entre instruções. Construíram operandos de comparação dentro dos jumps condicionais, não possui carry de operações aritméticas complicadas nem tem detector ou flag para erros aritméticos, incluindo overflow, underflow e divisão por 0 (zero).

9. Com o propósito de codificar as instruções inteiras para salva-las no espaço.
10. Um dos maiores destaque é na quantidade de espaço utilizado nas operações, ou seja o número de bits.
11. Não. Todas as condicionais são realizadas no comando *jump* que possui condicionais embutidos.
12. Com o intuído de facilitar a decodificação dos dados.
13. *Jump* salta apenas as instruções para um determinado endereço
    Jump and link salta as instruções para um determinado endereço, porem possui um endereço de retorno.
14. É um chip formado por 5 estágios que permitem gerar diferentes configurações de SoC.
15. O emulador C++ RTL é executado do diretório */emulator*. Neste diretório, um comando de execução *default* já á conhecido para verificar se a ferramenta foi instalada com sucesso. E para executa-lo um comando make é utilizado, iniciando assim a simulação.