

UNIVERSIDADE DE BRASÍLIA
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

116394 ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES

Trabalho I: Programação Assembler

OBJETIVO

Este trabalho objetiva a prática da programação em *assembler* do RISC-V. A partir de um código C que descreve parte do algoritmo de cifragem de dados IDEA, o aluno deve realizar sua tradução e implementação em assembler do RISC-V, utilizando o ambiente RARS para programação e teste do software.

DESCRIÇÃO

A. Algoritmo IDEA - Histórico

- Chamava-se inicialmente PES (Proposed Encryption Standard) e foi desenvolvido em 1990 por Xuejia Lai e James Massey
- Após uma demonstração de criptoanálise diferencial feita por Biham e Shamir, seus autores aprimoraram o algoritmo contra ataques e o chamaram de IPES (Improved Proposed Encryption Algorithm) (1991)
- Em 1992, o IPES teve seu nome mudado para IDEA (International Data Encryption Algorithm)

B. Algoritmo IDEA - Características

- Baseado em um novo conceito de “misturar operações de diferentes grupos algébricos”
- A requerida “confusão” (criptografia) é obtida pelo uso sucessivo de três operadores incompatíveis em pares de sub-blocos de 16 bits e a estrutura do algoritmo foi escolhida para prover a necessária “difusão”
- O algoritmo foi elaborado para facilitar implementações tanto em software quanto em hardware
- Algoritmo simétrico (mesma chave para cifrar e decifrar)
- Chave de 128 bits que gera 52 sub-chaves de 16 bits
- Opera em blocos de 64 bits dividido em sub-blocos de 16 bits
- Estrutura semelhante ao DES: possui um número fixo de iterações (rodadas) de uma mesma função que utiliza sub-chaves distintas, e o mesmo algoritmo serve para cifrar e decifrar alterando-se apenas a forma das gerações das sub-chaves e não há permutação de bits
- As operações realizadas em cada iteração são reversíveis

C. Algoritmo IDEA - As três operações básicas

- ou-exclusivo (XOR) sobre 16 bits
- soma $\text{mod } 2^{16}$, que é equivalente à soma usual em que o bit mais à esquerda correspondente ao valor 2^{16} deve ser sempre igual a zero após a soma
- multiplicação $\text{mod } (2^{16}+1)$, sendo que o valor 0 é representado por 2^{16} , para que a operação possa ser reversível

D. Algoritmo IDEA - Fluxograma

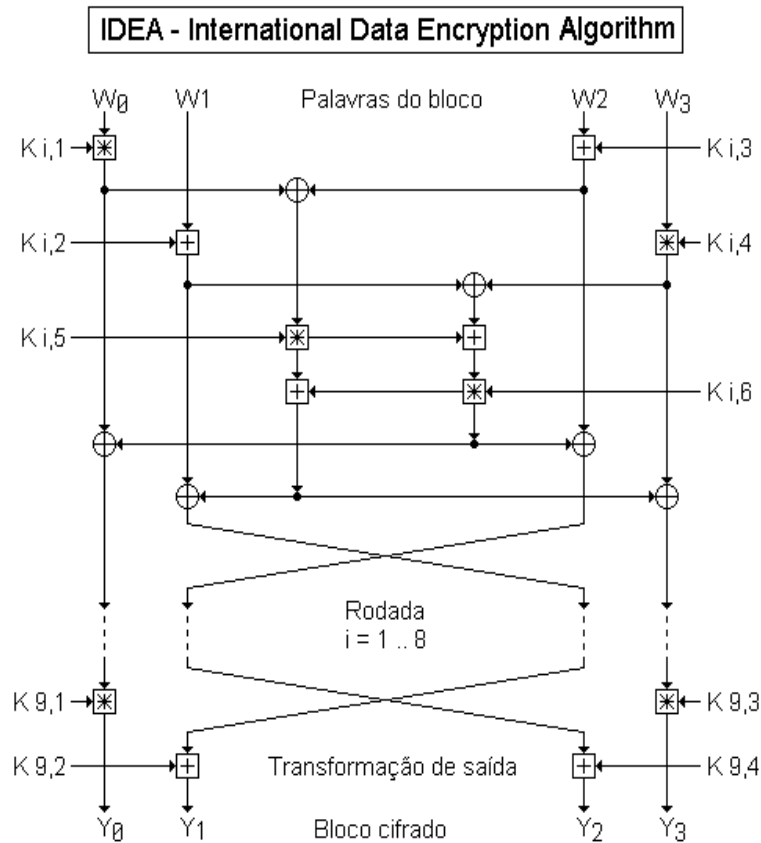


Figura 1. Fluxograma do IDEA

E. Algoritmo IDEA - Funções a serem implementadas

```
#include <iostream>

typedef unsigned short      uint16;           /* 16-bit word */
typedef unsigned int        uint32;           /* 32-bit word */

#define LSW16(y) ((y) & 0xffff)               /* low significant 16-bit */
#define MSW16(y) ((y >> 16) & 0xffff)         /* most significant 16-bit */

uint16 mul(uint16 x, uint16 y) {
    uint32 p = x*y;
    if (p == 0)
        x = 65537-x-y;
    else {
        x = p >> 16;
        y = p;
        x = y-x;
        if (y < x) x += 65537;
    }
    return x;
}

void idea_round(uint16 *blk_in_ptr, uint16 *blk_out_ptr, uint16 *key_ptr) {
    uint16 word1, word2, word3, word4;
    uint16 t1, t2;

    word1 = *blk_in_ptr++;
    word2 = *blk_in_ptr++;
    word3 = *blk_in_ptr++;
    word4 = *blk_in_ptr;
```

```

word1 = mul(word1, *key_ptr++);
word2 = LSW16(word2 + *key_ptr++);
word3 = LSW16(word3 + *key_ptr++);
word4 = mul(word4, *key_ptr++);

t2 = word1 ^ word3;
t2 = mul(t2, *key_ptr++);
t1 = LSW16(t2 + (word2 ^ word4));
t1 = mul(t1, *key_ptr++);
t2 = LSW16(t1 + t2);

word1 ^= t1;
word4 ^= t2;

t2 ^= word2;
word2 = word3 ^ t1;
word3 = t2;

*blk_out_ptr++ = word1;
*blk_out_ptr++ = word2;
*blk_out_ptr++ = word3;
*blk_out_ptr++ = word4;
}

int main (int argc, char * const argv[]) {

    uint16 blk_in[] =      { 0, 1, 2, 3 };
    uint16 blk_out[] =     { 0, 0, 0, 0 };
    uint16 keys[] =        { 1, 2, 3, 4, 5, 6 };

    idea_round(blk_in, blk_out, keys);

    for (int i = 0; i < 4; i++) {
        printf("Saída[%d] = %d\n", i, blk_out[i]);
    }

    return 0;
}

```

ENTREGA

Entregar no Moodle em um arquivo compactado:

- breve descrição do trabalho realizado
- descrição dos testes realizados para verificação do código assembler
- código assembler