

**UNIVERSIDADE FEDERAL DO MARANHÃO – UFMA**

**CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA – CCET**

**DISCIPLINA: ENGENHARIA ELÉTRICA**

NERVAL DE JESUS SANTOS JUNIOR

Discente

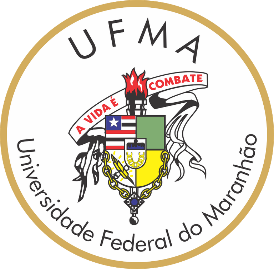
DENIVALDO LOPES

Docente Responsável

**RELATÓRIO DA UNIDADE II SOBRE HCS12**

São Luís - MA

2023



NERVAL DE JESUS SANTOS JUNIOR

Discente

**RELATÓRIO DA UNIDADE II SOBRE HCS12**

Relatório apresentado(a) para disciplina **DEEE0187 – INTROD.À ARQUITET. DE COMPUTADORES (EE) (2023.1-T01)** da Universidade Federal do Maranhão – UFMA, ministrada pelo(a) docente doutor(a), Denivaldo Lopes.

São Luís - MA

2023

**Sumário**

[1. Introdução 3](#_heading=h.2s8eyo1)

[2. Objetivos 4](#_heading=h.17dp8vu)

[3. Materiais 5](#_heading=h.3rdcrjn)

[4. Procedimentos 6](#_heading=h.26in1rg)

[5. Resultados e Discussões 13](#_heading=h.lnxbz9)

[6. Conclusão 14](#_heading=h.35nkun2)

[7. Referências Bibliográficas 16](#_heading=h.1ksv4uv)

# Introdução

Nos últimos anos, os microcontroladores têm desempenhado um papel fundamental no avanço da tecnologia, encontrando aplicações em diversos setores, desde a indústria automotiva até dispositivos médicos. Entre os microcontroladores amplamente utilizados, o HCS12 se destaca por sua versatilidade e poder de processamento. Com sua arquitetura robusta e conjunto abrangente de recursos, o HCS12 oferece um ambiente propício para o desenvolvimento de sistemas embarcados.

Este trabalho tem como objetivo explorar as funcionalidades e características do microcontrolador HCS12, bem como apresentar soluções para problemas encontrados durante o desenvolvimento de projetos utilizando o ambiente de desenvolvimento Codewarrior.

O microcontrolador HCS12 é um dispositivo eletrônico que tem sido amplamente utilizado em sistemas embarcados, como automação industrial, robótica e sistemas automotivos. Para explorar suas capacidades, é necessário utilizar um ambiente de desenvolvimento integrado (IDE) como o Codewarrior. No entanto, durante o processo de desenvolvimento, é comum encontrar problemas que precisam ser solucionados para garantir o bom funcionamento do sistema. Neste trabalho, serão exploradas as capacidades do microcontrolador HCS12 e serão apresentadas implementações que solucionam problemas no ambiente Codewarrior. O objetivo é demonstrar como é possível utilizar o HCS12 em sistemas embarcados e como solucionar problemas no ambiente de desenvolvimento. Algumas fontes que podem ser utilizadas para a realização deste trabalho são o manual do usuário do HCS12() e a comunidade NXP , que oferece suporte e soluções para problemas relacionados ao Codewarrior.

O Codewarrior é uma poderosa plataforma de desenvolvimento integrado (IDE) projetada especificamente para atender às necessidades dos desenvolvedores de sistemas embarcados. Ele oferece uma interface intuitiva e uma ampla gama de recursos para facilitar o processo de desenvolvimento, como compiladores, depuradores e emuladores. No entanto, assim como em qualquer ambiente de programação, os desenvolvedores podem se deparar com desafios e problemas ao utilizar o Codewarrior.

Durante o decorrer deste trabalho, serão abordados diferentes aspectos do microcontrolador HCS12, desde sua arquitetura e recursos até a implementação de soluções práticas para problemas encontrados no Codewarrior. Serão exploradas técnicas e estratégias para lidar com erros de compilação, depuração de código, otimização de desempenho e integração de bibliotecas externas.

Ao final deste trabalho, espera-se que o leitor tenha adquirido um entendimento aprofundado sobre o microcontrolador HCS12, suas características-chave e possíveis soluções para os desafios encontrados no ambiente Codewarrior. Além disso, espera-se que os exemplos práticos apresentados inspirem e auxiliem outros desenvolvedores a explorarem ainda mais as capacidades do HCS12 e a aprimorarem suas habilidades de programação em sistemas embarcados.

O objetivo deste relatório é apresentar de forma clara e objetiva os procedimentos e programas que foram desenvolvidos em aula prática, dentro da disciplina de laboratório de arquitetura de computadores. Será feita também a exposição dos resultados obtidos com o experimento e uma conclusão geral a respeito do trabalho realizado.

# Metodologia

* A nota será baseada na solução dos problemas postos, incluindo execução, teste e análise dos programas feitos em *assembly* e/ou C para HCS12 (conforme especificado em cada questão); - Os estudantes de equipes diferentes não podem trocar informações sobre as soluções, pois havendo cópias ou semelhanças fortes, os trabalhos serão desconsiderados;
* O trabalho deve ser fornecido em documento no formato PDF, contendo capa, sumário, introdução, metodologia, desenvolvimento das questões (incluindo fluxograma, código em *assembly*/linguagem C), conclusão e bibliografia (formato ABNT ou IEEE). O código fonte de cada questão deve ser enviado em arquivo compactado. Coloquem o nome dos membros da equipe em cada código fonte;

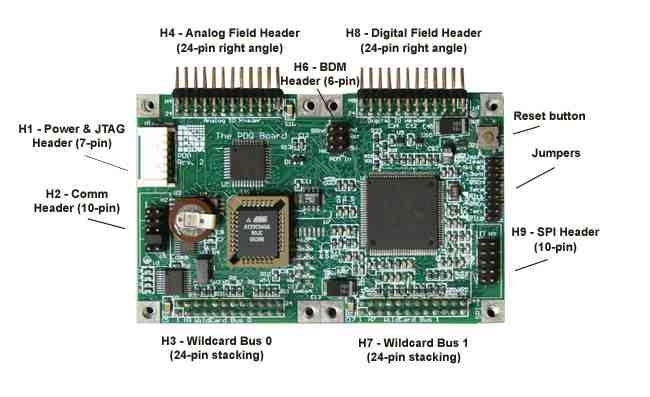
Para a realização adequada do experimento e compreensão do mesmo, serão seguidos os seguintes passos:

1. Primeiramente será utilizado o *software* Logisim, com ele será implementada a tabela-verdade referente ao problema ao qual será feita a solução.
2. Com a tabela pronta, será gerada, através de minimização por mapa de Karnaugh, a expressão booleana correspondente.
3. Com o auxílio da ferramenta online, Draw.io, será feito o diagrama de blocos que servirá como modelo para construção do programa que será executado no microcontrolador.
4. O *software* computacional CodeWarrior será o ambiente de programação, na linguagem C, para implementação do algoritmo de controle.

O programa será carregado e executado no microcontrolador HCS12, a fim de ser testado e observado os resultados finais.

O CodeWarrior Development Studio é um IDE (Ambiente de Desenvolvimento) completo e integrado que fornece uma estrutura altamente visual e automatizada para acelerar o desenvolvimento dos aplicativos incorporados mais complexos.

As soluções de software e desenvolvimento da NXP levam você além do silício, ajudando a dar vida a seus aplicativos incorporados. O CodeWarrior Development Studio é um IDE (Ambiente de Desenvolvimento) completo e integrado que fornece uma estrutura altamente visual e automatizada para acelerar o desenvolvimento dos aplicativos incorporados mais complexos.



# Objetivos

**Objetivos Específicos**

# Materiais

Nesse experimento usamos os seguintes materiais:

* **Hardware:**
* - Microcomputador (processador Core i5-XXXX, RAM 8 GB, HD 1TB, Monitor 17”);
* - Kit de desenvolvimento para HCS12: APS12C128SLK [Ref1];
* - Kit de desenvolvimento para HCS12 (placa de expansão): SLK0109UG [Ref2];
* **Software:**
* - Sistema operacional: Windows 7 64 bits;
* - Ambiente de desenvolvimento integrado (IDE): CodeWarrior versão 5.1;

# Procedimentos

Questao 1) Pede-se que seja feito um programa em *assembly* para HCS12 que corresponda ao programa em linguagem de alto nível a seguir. (Pontos: 3,33)

OBS: Utilizar instruções para manipular valores inteiros de 16 bits (1 word).

**Programa em linguagem de alto nível**

//Questão 1 Pede-se que seja feito um programa em assembly para HCS12 que corresponda ao programa em linguagem de alto nível a seguir: Programa em linguagem de alto nível

void main(){

int X = readPortB();     // X é uma variável que armazena valores inteiros lidos da PortaB

int Y = -3;

int Z = 7;

int W=0;

int R=0;

int T=0;

int K=0;

int S=0;

R = 3\*X + 2 \* Y + Z;

    T =   R +2\*X - Z;       K = 2\*(X + Z);

if(K  <= T){

        S = 2 \* Y + X;

           W= funcA(S,R);

    }else {

        S = funcB(X,T);

        W = funcC(X,S,T);

    }

}

int funcA(int *parA1*, int *parA2*){

      return 3\* *parA1* + *parA2*;

}

int funcB(int *parB1*, int *parB2*){

        return *parB1* +  *parB2* + 3;

}

int funcC(int *parC1*, int *parC2*, int *parC3*){

        return 2\**parC1* + *parC2* - *parC3*;

}

codigo em linguagem c- IDE do vscode

Questao 3) Elabore um programa em C para HCS12 que leia um valor short (8 bits) em complemento de dois pela porta A. Depois, calcule o fatorial deste valor e o apresente na porta B. (Pontos: 3,33)

#include <hidef.h>      /\* common defines and macros \*/

#include "derivative.h"      /\* derivative-specific definitions \*/

#include "Cpu.h"

#include "Events.h"

#include "A.h"

#include "B.h"

#include "PE\_Types.h"

#include "PE\_Error.h"

#include "PE\_Const.h"

#include "IO\_Map.h"

void main(void) {

  /\* put your own code here \*/

      /\* Write your local variable definition here \*/

    short valueA;

    short factorial = 1;

    /\*\*\* Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! \*\*\*/

    PE\_low\_level\_init();

    /\*\*\* End of Processor Expert internal initialization.                    \*\*\*/

    /\* Write your code here \*/

    valueA = A\_GetVal(); // Ler o valor da porta A

    if (valueA < 0) {

        // Converter o valor em complemento de dois para valor positivo

        valueA = valueA & 0xFF; // Limpar os bits de sinal (mais significativo)

        valueA = -valueA; // Inverter o sinal

    }

    // Calcular o fatorial do valor

    for (int i = 1; i <= valueA; i++) {

        factorial \*= i;

    }

    B\_PutVal(factorial); // Enviar o valor do fatorial para a porta B

    /\*\*\* Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! \*\*\*/

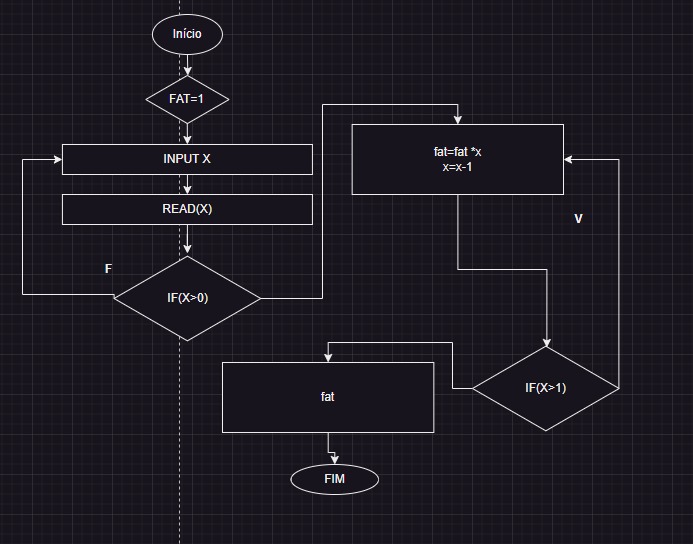
  for(;;) {

    \_FEED\_COP(); /\* feeds the dog \*/

  } /\* loop forever \*/

  /\* please make sure that you never leave main \*/

}



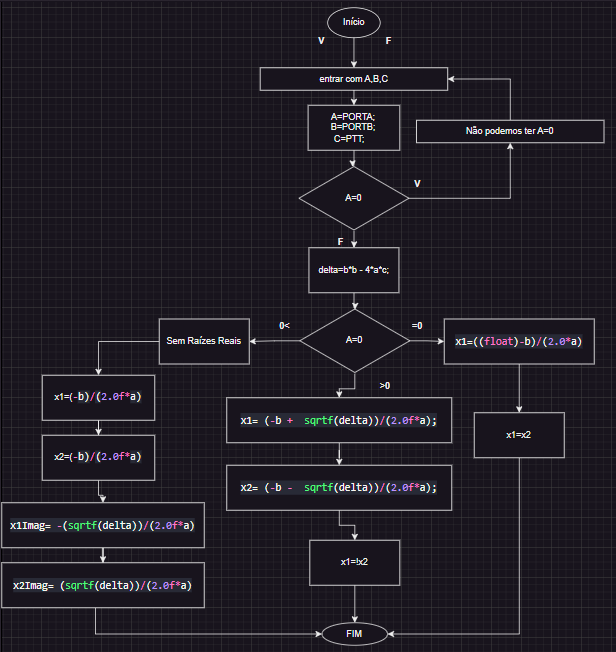
Questao 4) Um sistema de software/hardware baseado no HCS12 deve obter as raízes de uma equação do 2º grau, dada por *a\*x2+b\*x+c=0*, onde *a* é lido pela Porta A, *b* é lido pela porta B e *c* é lido pela Porta T. Os valores de *a*, *b* e *c* são lidos como valores inteiros em complemento de dois. As raízes desta equação devem ser armazenadas em variáveis denominadas *x1* e *x2*. Elabore este programa em linguagem C. (Pontos: 3,33)

Resposta :

* O sistema de software/hardware é baseado no HCS12.
* A equação do 2º grau, ax2+bx+c=0, deve ser resolvida.
* Os valores dos coeficientes a, b e c são lidos pelas portas A, B e T, respectivamente.
* Os valores lidos são inteiros em complemento de dois.
* As raízes da equação devem ser armazenadas nas variáveis x1 e x2.
* O programa deve ser elaborado em linguagem C.

Quanto a uma tabela, você pode criar uma tabela mostrando os valores lidos pelos registradores de porta A, B e T, e os valores calculados das raízes x1 e x2. Algo assim:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tabela da Verdade da quarta Questão** | | | | |
| **Porta A** | **Porta B** | **Porta T** | **X1** | **X2** |
| **A** | **B** | **C** | **saida x1** | **saida x22** |
| leitura da porta | leitura da porta | leitura da porta | raiz 1 | raiz 2 |



/\*\* ###################################################################

\*\*     Filename  : HelloWorld\_C.c

\*\*     Project   : HelloWorld\_C

\*\*     Processor : MC9S12C128CFU16

\*\*     Version   : Driver 01.14

\*\*     Compiler  : CodeWarrior HC12 C Compiler

\*\*     Date/Time : 30/06/2023, 18:30

\*\*     Abstract  :

\*\*         Main module.

\*\*         This module contains user's application code.

\*\*     Settings  :

\*\*     Contents  :

\*\*         No public methods

\*\*

\*\* ###################################################################\*/

/\* MODULE HelloWorld\_C \*/

//meu nome é Nerval Junior

/\* Including needed modules to compile this module/procedure \*/

#include "Cpu.h"

#include "Events.h"

/\* Include shared modules, which are used for whole project \*/

#include "PE\_Types.h"

#include "PE\_Error.h"

#include "PE\_Const.h"

#include "IO\_Map.h"

/\* User includes (#include below this line is not maintained by Processor Expert) \*/

#include <math.h>

float x1, x2,delta;

float a,b,c;

float x1Imag, x2Imag;

void main(void)

{

  /\* Write your local variable definition here \*/

  /\*\*\* Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! \*\*\*/

  PE\_low\_level\_init();

  /\*\*\* End of Processor Expert internal initialization.                    \*\*\*/

  /\* Write your code here \*/

  x1=0;

  x2=0;

  delta=0;

  x1Imag=0;

  x2Imag=0;

  a=PORTA;

  b=PORTB;

  c=PTT;

  // função do delta da equação do segundo grau

  delta=b\*b - 4\*a\*c;

  if(delta ==0){

  // raizes reais iguais

      x1=((float)-b)/(2.0\*a);

      x2=x1;

  }else if(delta > 0){

      x1= (-b +  sqrtf(delta))/(2.0f\*a);

      x2= (-b -  sqrtf(delta))/(2.0f\*a);

  }else{

      x1= (-b)/(2.0f\*a);

      x2= (-b)/(2.0f\*a);

      x1Imag= -(sqrtf(delta))/(2.0f\*a);

      x2Imag= (sqrtf(delta))/(2.0f\*a);

  }

  /\*\*\* Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! \*\*\*/

  for(;;){}

  /\*\*\* Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! \*\*\*/

  } /\*\*\* End of main routine. DO NOT MODIFY THIS TEXT!!! \*\*\*/

/\* END HelloWorld\_C \*/

/\*

\*\* ###################################################################

\*\*

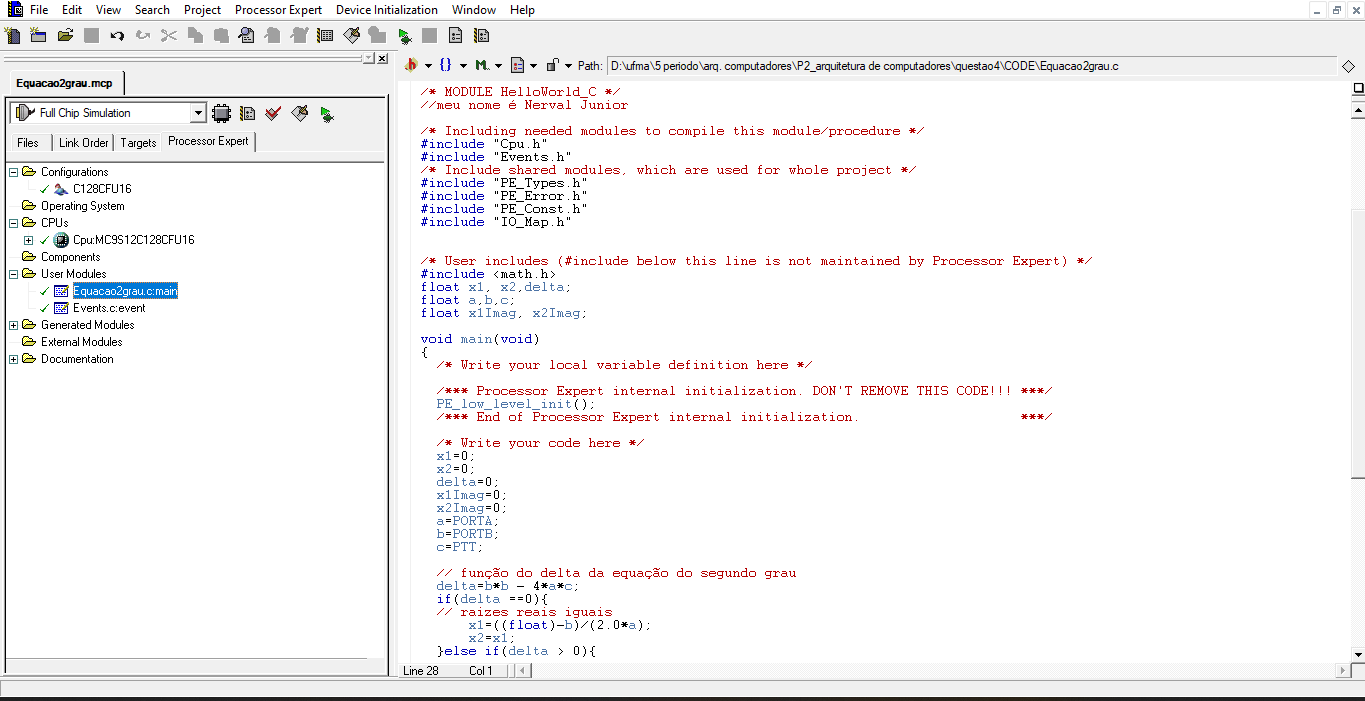
\*\*     This file was created by Processor Expert 3.05 [04.46]

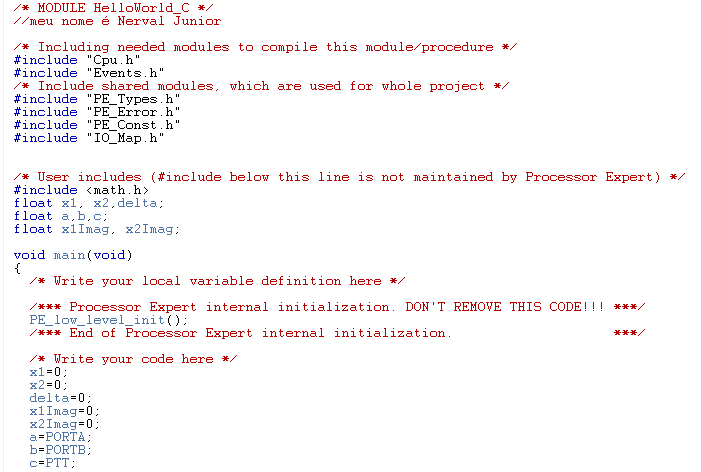
\*\*     for the Freescale HCS12 series of microcontrollers.

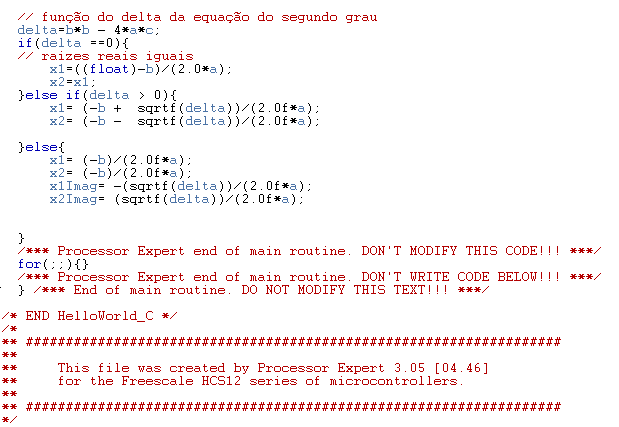
\*\*

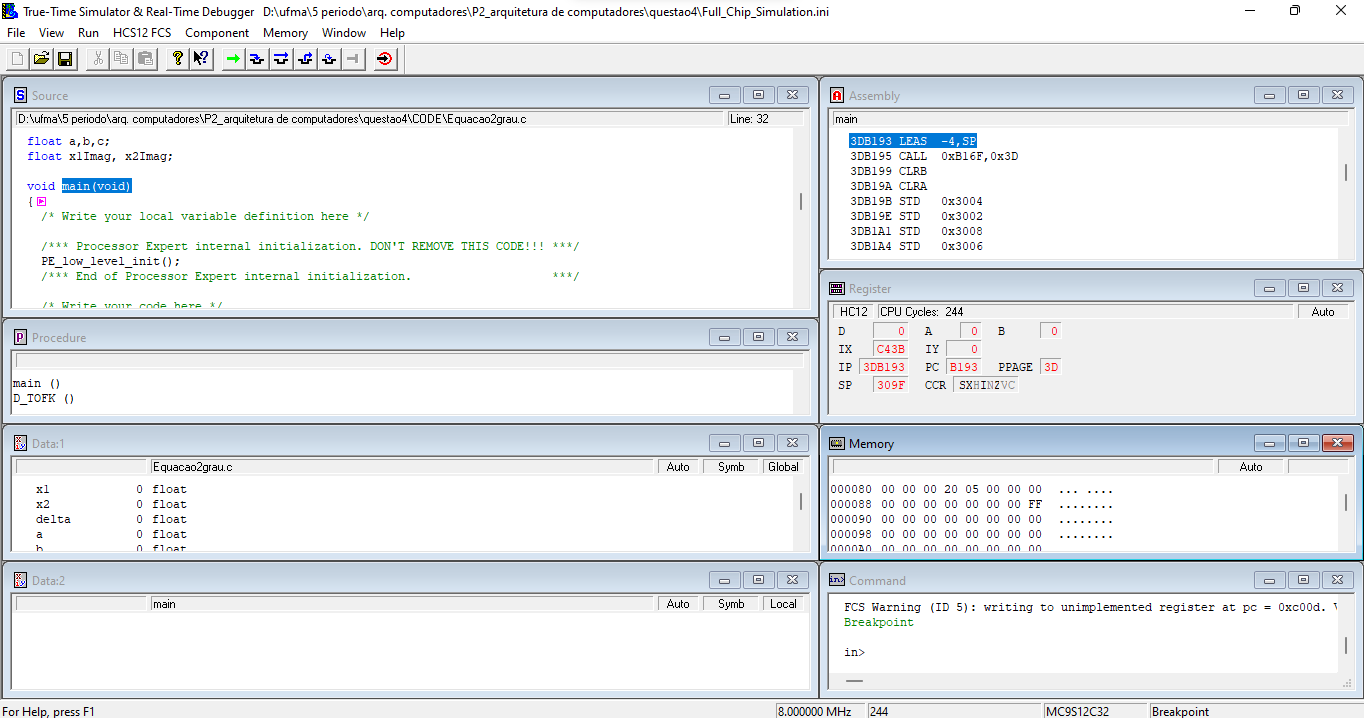
\*\* ###################################################################

\*/









Questao 5) Escreva um programa em linguagem C para o HCS12 que faça o controle da seguinte planta. (Pontos: 3,33)

D

I

C

B

A

E

V

S

E

VE

Controlador

(

HCS12

)

Eletroválvula de saída

Eletroválvula de entrada

Um reservatório para armazenar um determinado produto líquido deve ser controlado da seguinte forma:

* + - Ao pressionar o botão *I*, o sistema deve ser acionado;
    - Uma vez que o sistema tenha sido acionado, a eletroválvula *Evs* deve ser ligada se, e somente se, o líquido estiver no nível A ou nível B. Se o líquido estiver no nível D, então a eletroválvula *Evs* deve ser desligada até que o nível B seja atingido;
    - A eletroválvula *Eve* deve ser ligada se, e somente se, o botão *I* estiver ligado e o nível do líquido estiver no nível D ou C ou B. No nível A, a eletroválvula *Eve* deve ser desligada;
    - Caso o sistema esteja ligado e seja pressionado o botão *I* para desligá-lo, então a Eletroválvula *Eve* deve ser desligada. Em seguida, todo o líquido deve ser escoado pela Eletroválvula *Evs*, até que o sensor D acuse reservatório vazio (sensor desativado).

OBS: Utilize os seguintes bits das portas:

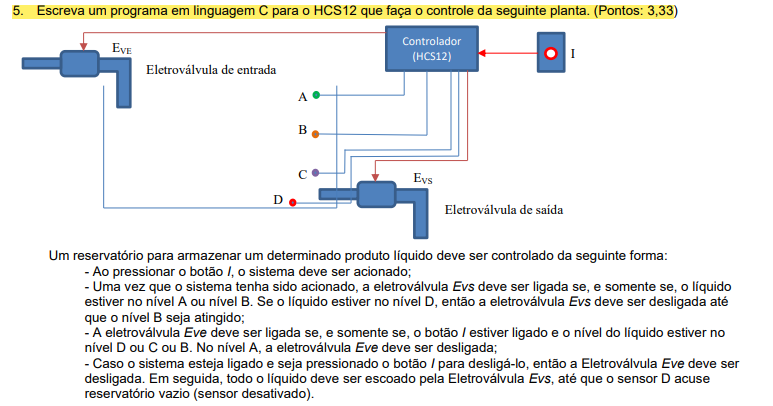
* + - PB0 como entrada para I;
    - PP0 como entrada para A;
    - PB1 como entrada para B;
    - PB2 como entrada para C;
    - PB3 como entrada para D;
    - PB4 como saída para Evs;
    - PB5 como saída para Eve;
    - PB7 como sinal de alerta para mau funcionamento do sistema, por exemplo: sensor A e B ativos, mas sensor C desativado; ou sensor A ativado, mas sensores B e C desativados; ou B ativado, mas sensor C desativado.

OBS: Faça a tabela da verdade, depois obtenha a expressão simplificada pelo mapa de Karnaugh, faça o fluxograma da solução do problema, escreva o programa em linguagem C para HCS12.

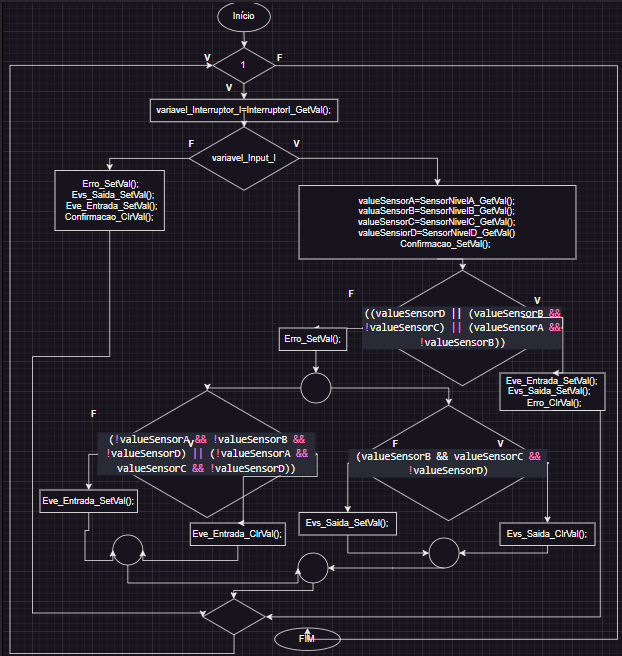
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabela da Verdade da quinta Questão** | | | | | | | |
| **PB0(I)** | **PB3(D)** | **PB2(C)** | **PP1(B)** | **PP0(A)** | **PB4(Evs)** | **PB5(Eve)** | **PB7(Erros)** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |

# Resultados e Discussões

O primeiro experimento consiste em fazer o controle de um tanque de produtos químicos conforme o enunciado abaixo:

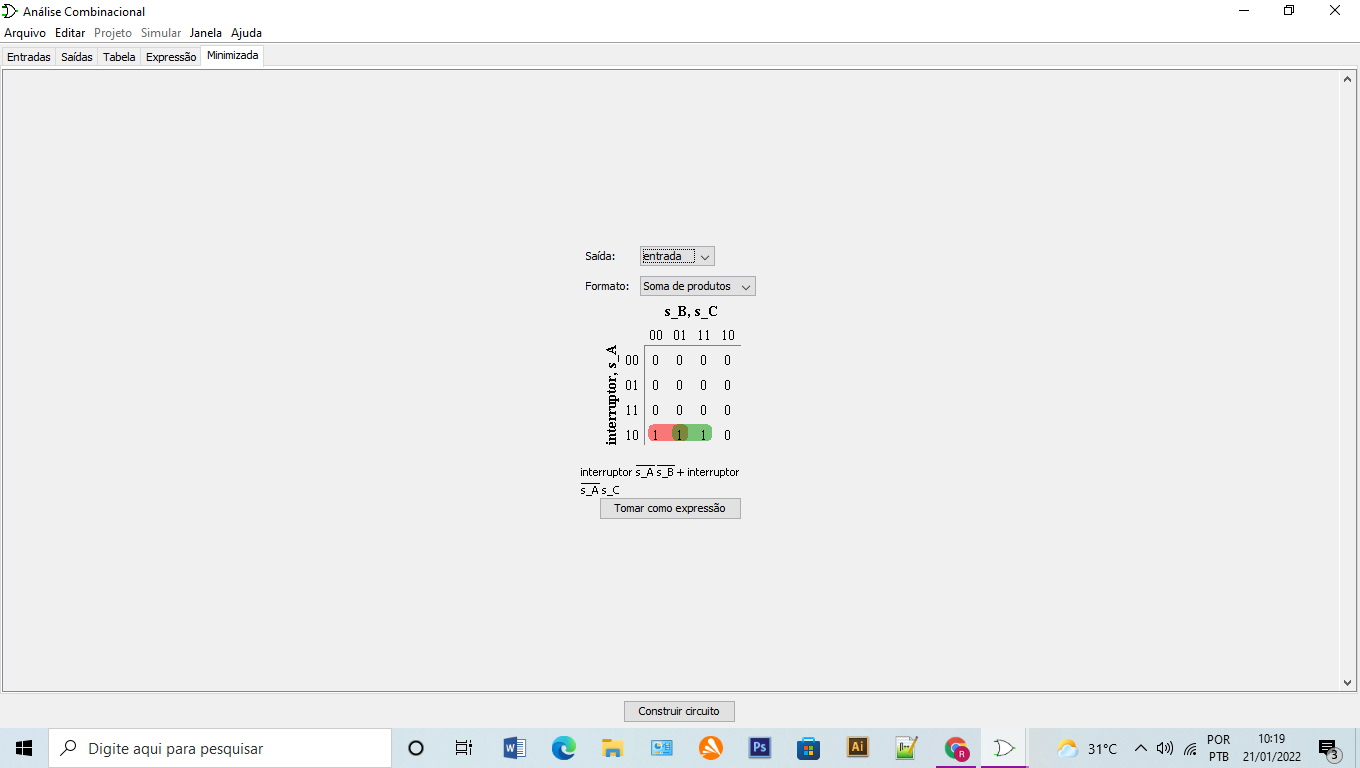


1. Tabela-Verdade para o problema:

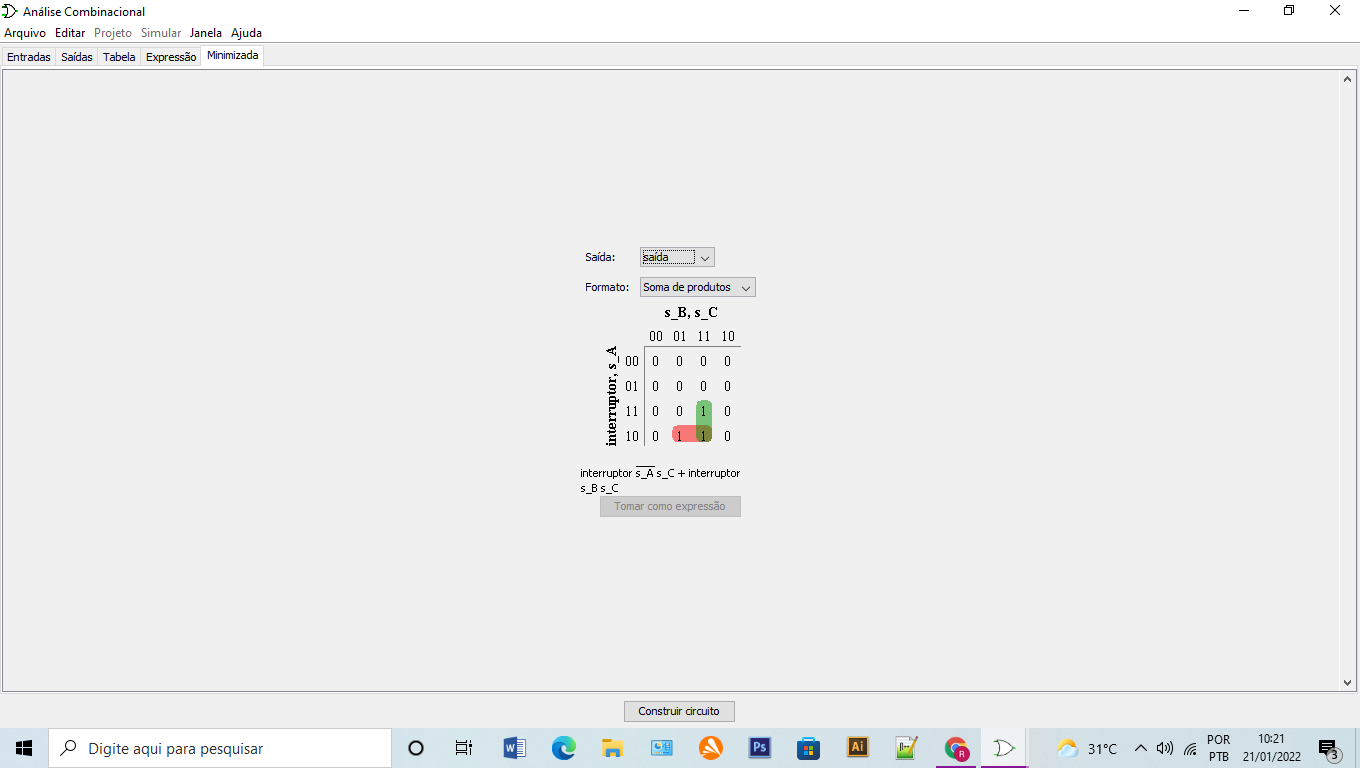


1. Expressões extraídas através do mapa de Karnaugh:

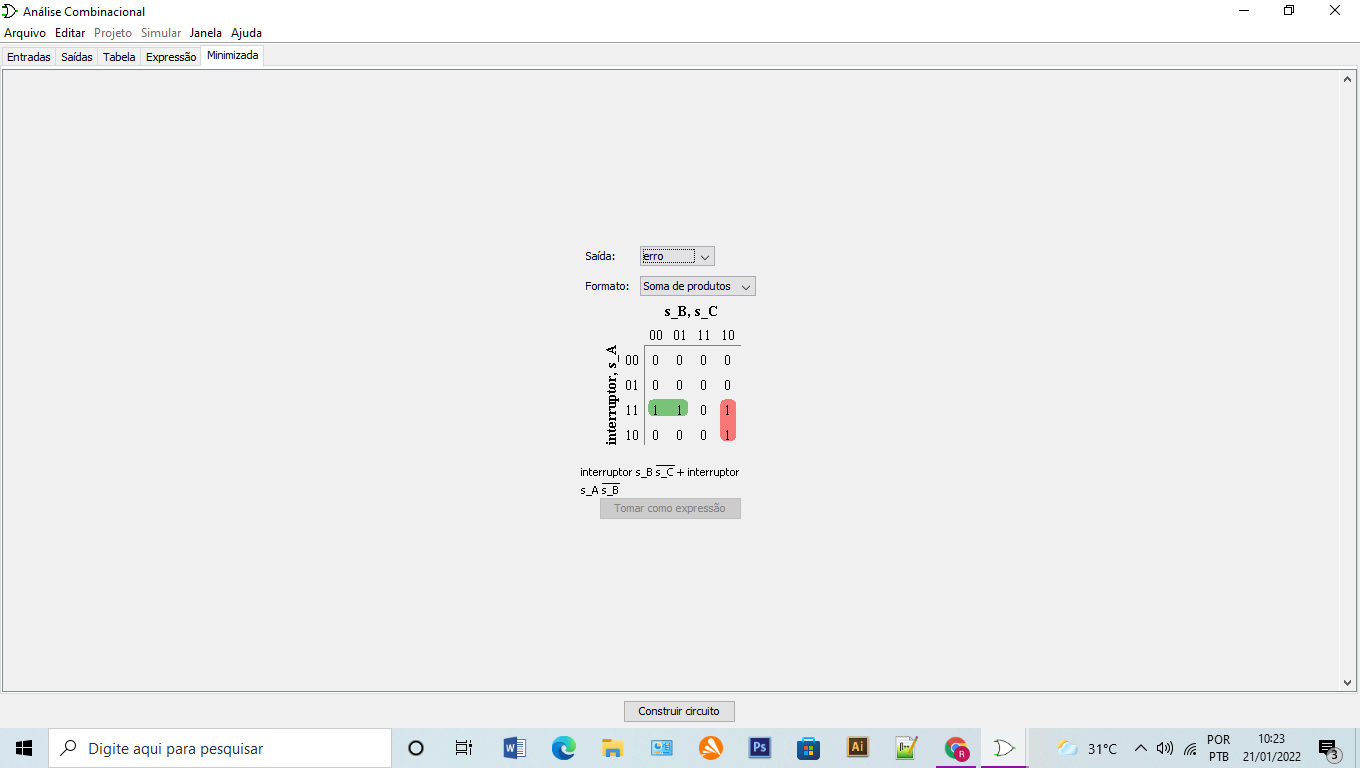
**Entrada:**

****

**Saída:**

****

**Erro:**

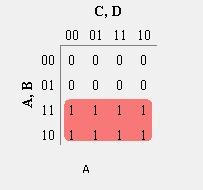
****

1. Com a tabela e as expressões geradas, pode-se gerar o diagrama de blocos correspondente:
2. O código em linguagem C será o seguinte:

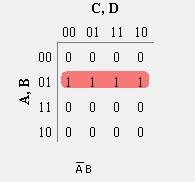
Depois o código foi executado no microcontrolador e os resultados foram apurados.

1. Tabela-Verdade para o problema:
2. Expressões obtidas através do mapa de Karnaugh:

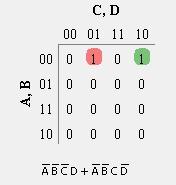
**PÁTIO A:**



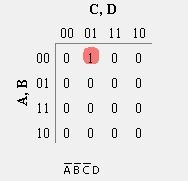
**PÁTIO B:**



**PÁTIO C:**



**PÁTIO D:**



1. Com a tabela e as expressões geradas, pode-se gerar o diagrama de blocos correspondente:

# Conclusão

Através dos experimentos foi possível compreender de forma clara e prática os conhecimentos adquiridos de forma teórica em outras disciplinas. Todos os passos seguidos e as dificuldades para concluir os projetos serviram de grande aprendizado de como é o dia a dia do engenheiro eletricista e como deve-se agir diante de tais situações.

Com os experimentos trazidos pelo professor ficou evidente as diversas aplicações de microcontroladores no cotidiano de diversas empresas e como é possível aproveitar situações vivenciadas na indústria para implementar tecnologias que auxiliem um usuário e dê a ele uma maior praticidade para realizar suas tarefas.

# Referências Bibliográficas

[1] Draw.io: Software para construção de diagrama de blocos. Disponível em: https://draw.io. Acesso em: 21 jan. 2023.

[2] NXP. CodeWarrior Development Studio para Microcontroladores HCS12 (Classic IDE) versão 5.2, 2016.

[3] Logisim: Software de simulação de circuitos digitais. Versão estável: 2.7.1, 2011; Disponível em:<http://www.cburch.com/logisim/pt/index.html> . Acesso em: 21 jan 2023.

HCS12 Microcontroller and Embedded Systems: Using Assembly and C for Freescale HCS12. Muhammad Ali Mazidi, Janice Gillispie Mazidi, Rolin D. McKinlay.

NXP Community. Disponível em: https://community.nxp.com/. Acesso em 01 de julho de 2023.