

**UNIVERSIDADE FEDERAL DO MARANHÃO – UFMA**

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

**DISCIPLINA: ENGENHARIA ELÉTRICA**

NERVAL DE JESUS SANTOS JUNIOR

Discente

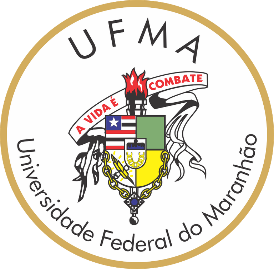
DENIVALDO LOPES

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 4](#_heading=h.2s8eyo1)

[**2. Metodologia 5**](#_heading=)

[3. Objetivos 5](#_heading=h.17dp8vu)

[4. Materiais 6](#_heading=h.3rdcrjn)

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

[6. Resultados e Discussões 24](#_heading=h.lnxbz9)

[7. Conclusão 27](#_heading=h.35nkun2)

[Referências Bibliográficas 28](#_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. No livro "Embedded Systems: Introduction to ARM Cortex-M Microcontrollers" de Valvano 2012, é apresentada uma introdução abrangente sobre microcontroladores, destacando o HCS12 como um componente central no campo dos 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 oferecem 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.

# Metodologia

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.
5. O programa será carregado e executado no microcontrolador HCS12, a fim de ser testado e observado os resultados finais.
6. 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).

# Objetivos

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.

**Objetivos Específicos**

# Materiais

Nesse experimento usamos os seguintes materiais:

* **Hardware:**
* - Microcomputador (processador Core i7, RAM 8GB, HD 1TB, Monitor 16”);
* - Kit de desenvolvimento para HCS12: APS12C128SLK [Ref1];
* - Kit de desenvolvimento para HCS12 (placa de expansão): SLK0109UG [Ref2];
* **Software:**
* - Sistema operacional: Windows 10 64 bits;
* - Ambiente de desenvolvimento integrado (IDE): CodeWarrior versão 5.1;

# Procedimentos

Questão 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*;

}

código em linguagem c- IDE do vscode

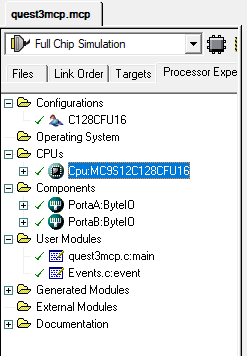
Nessa questão pede-se para ser criado um programa dentro do codeWarrior que faça os processos de acordo com o descrito na linguagem de alto nível.

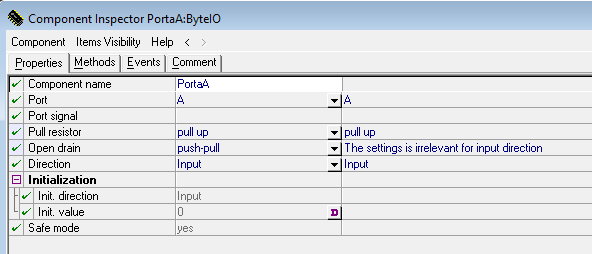
Primeiramente, foi realizada a análise do código de maneira a criar um fluxograma que expressasse o código em assembly. Depois de fazer essa análise é necessário criar o arquivo com todas as dependências que o codewarrior oferece para a simulação. Depois desse processo faz-se necessário desenvolver a mesma lógica dentro do arquivo (.asm) que tem a o escopo inicial da estrutura do código em assembly. Então foi necessário fazer a criação de alguns componentes que como a porta B, para ser realizada a leitura da porta depois da compilação.

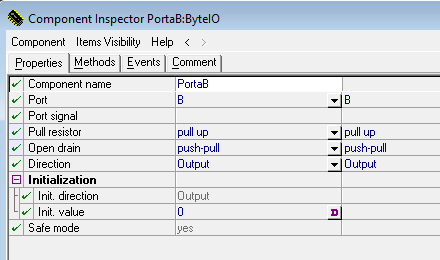
Foi de fundamental importância os arquivos sobre HCS12 que o professor mandou para funcionar como base. O Datasheet foi muito útil principalmente para entendermos a estrutura eletrônica do HSC12 e poder aplicar de acordo com os comandos em assembly.

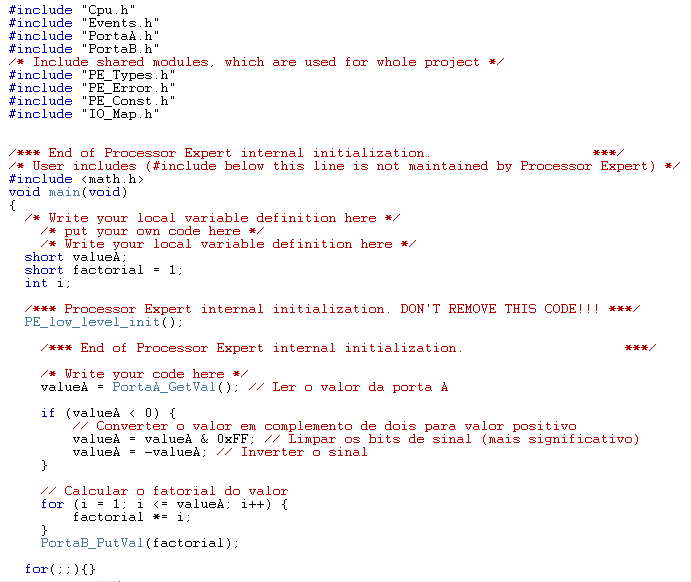
Questão 3) Elabore um programa em C para HCS12 que leia um valor short (8 bits) em complemento de dois pela porta A. Depois, calcule o valor fatorial deste valor e o apresente na porta B. (Pontos: 3,33)

| **Tabela da Verdade da terceira Questão** | |
| --- | --- |
| **valor (A)** | **fatorial (A!)** |
| 0 | 1 |
| 1 | 1 |
| 2 | 2 |
| 3 | 6 |
| 4 | 24 |
| 5 | 120 |
| 6 | 720 |
| 7 | 5040 |
| 8 | 40320 |
| 9 | 362880 |
| 10 | 3628800 |









**Programa em linguagem de alto nível**

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

\*\* Filename : quest3mcp.c

\*\* Project : quest3mcp

\*\* Processor : MC9S12C128CFU16

\*\* Version : Driver 01.14

\*\* Compiler : CodeWarrior HC12 C Compiler

\*\* Date/Time : 01/07/2023, 21:00

\*\* Abstract :

\*\* Main module.

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

\*\* Settings :

\*\* Contents :

\*\* No public methods

\*\*

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

/\* MODULE quest3mcp \*/

/\*

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

\*/

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

#include "Cpu.h"

#include "Events.h"

#include "PortaA.h"

#include "PortaB.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"

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

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

#include <math.h>

void main(void)

{

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

/\* put your own code here \*/

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

short valueA;

short factorial = 1;

int i;

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

PE\_low\_level\_init();

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

/\* Write your code here \*/

valueA = PortaA\_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 (i = 1; i <= valueA; i++) {

factorial \*= i;

}

PortaB\_PutVal(factorial);

for(;;){}

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

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

/\* END quest3mcp \*/

/\*

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

\*\*

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

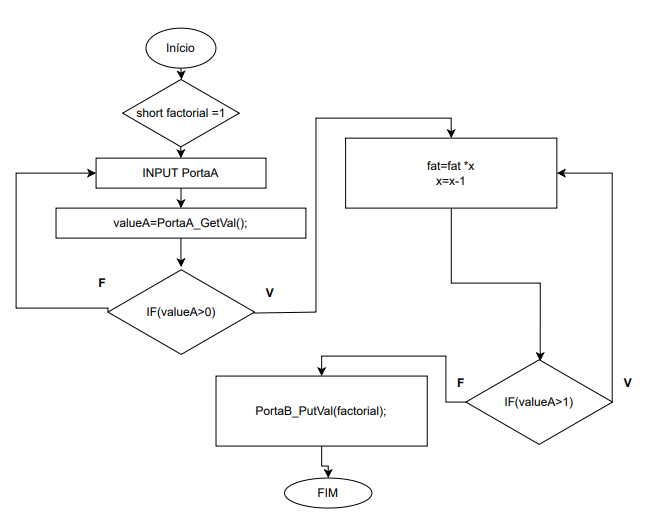
\*\* for the Freescale HCS12 series of microcontrollers.

\*\*

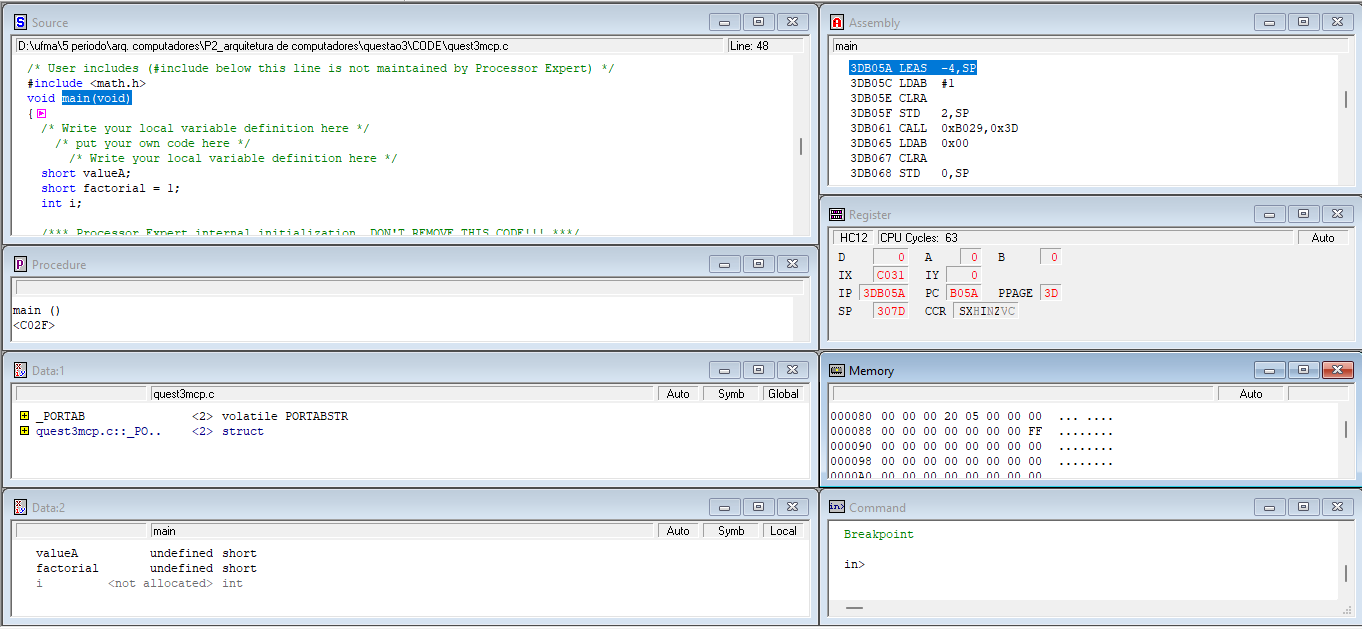
\*\* ###################################################################

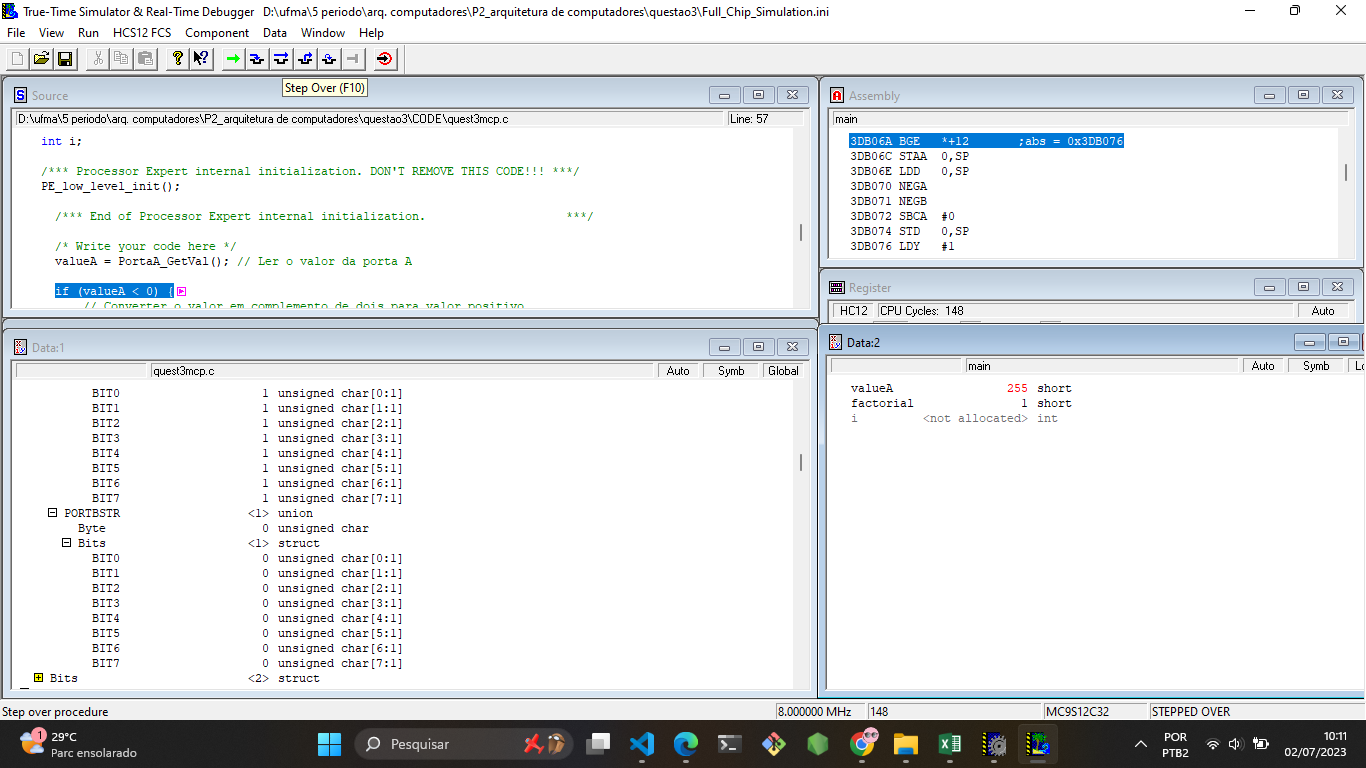
\*/

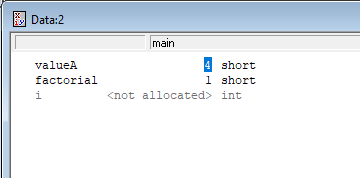
código em linguagem c- IDE do vscode

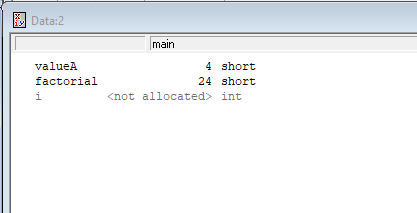
Figura X- Fluxograma questão 3

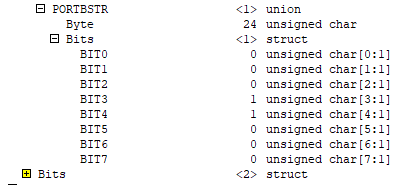
Fonte: Autoria Própria











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.

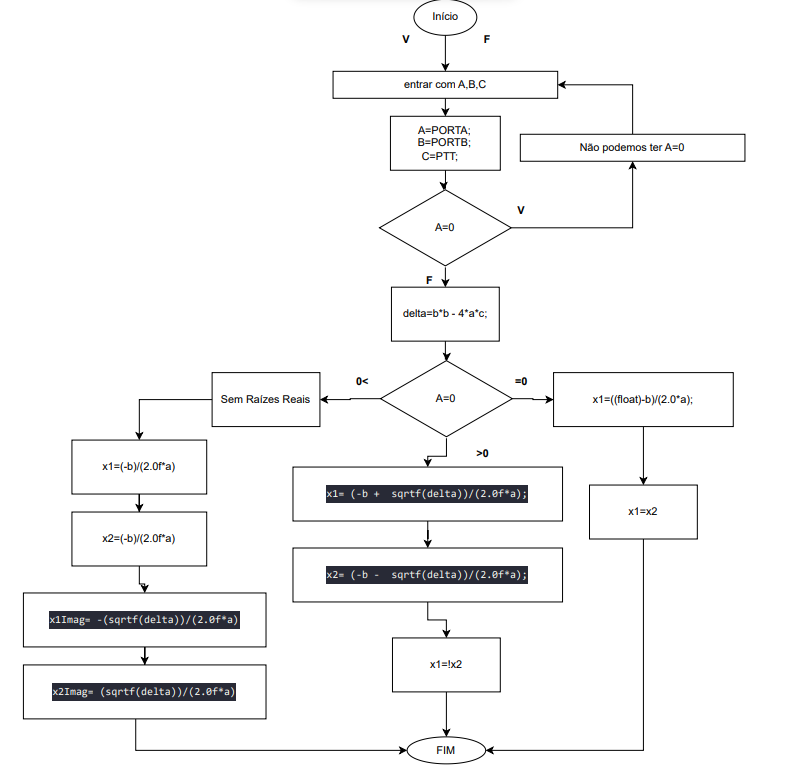
Quanto a uma tabela, você pode criar uma tabela mostrando os valores lidos pelos registradores de porta A, B e T, e os valores calculados das raízes x1 e x2. Algo assim:

Tabela X - Tabela verdade questão 4

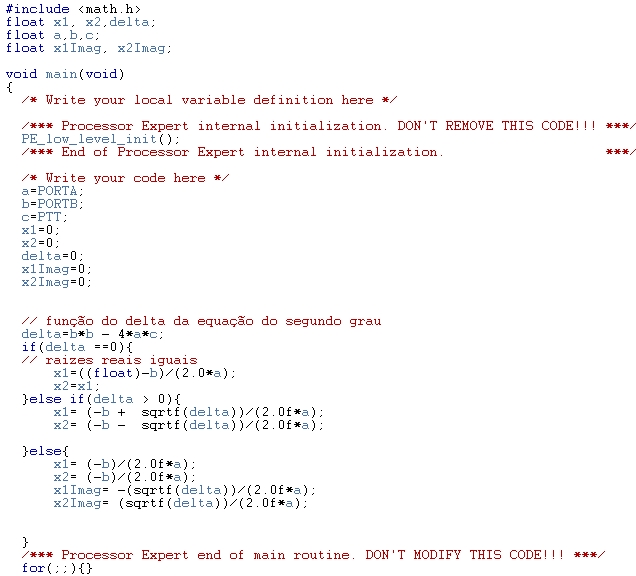
| **Tabela da Verdade da quarta Questão** | | | | |
| --- | --- | --- | --- | --- |
| **Porta A** | **Porta B** | **Porta T** | **X1** | **X2** |
| **A** | **B** | **C** | **saida x1** | **saida x22** |
| leitura da porta | leitura da porta | leitura da porta | raiz 1 | raiz 2 |

Fonte: Autoria própria

Figura X- Fluxograma questão 4



Fonte: Autoria própria



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

\*\* 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 \*/

/\*

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

\*/

/\* 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 \*/

a=PORTA;

b=PORTB;

c=PTT;

x1=0;

x2=0;

delta=0;

x1Imag=0;

x2Imag=0;

// 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 \*/

/\*

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

\*\*

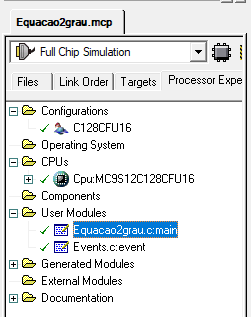
\*\* This file was created by Processor Expert 3.05 [04.46]

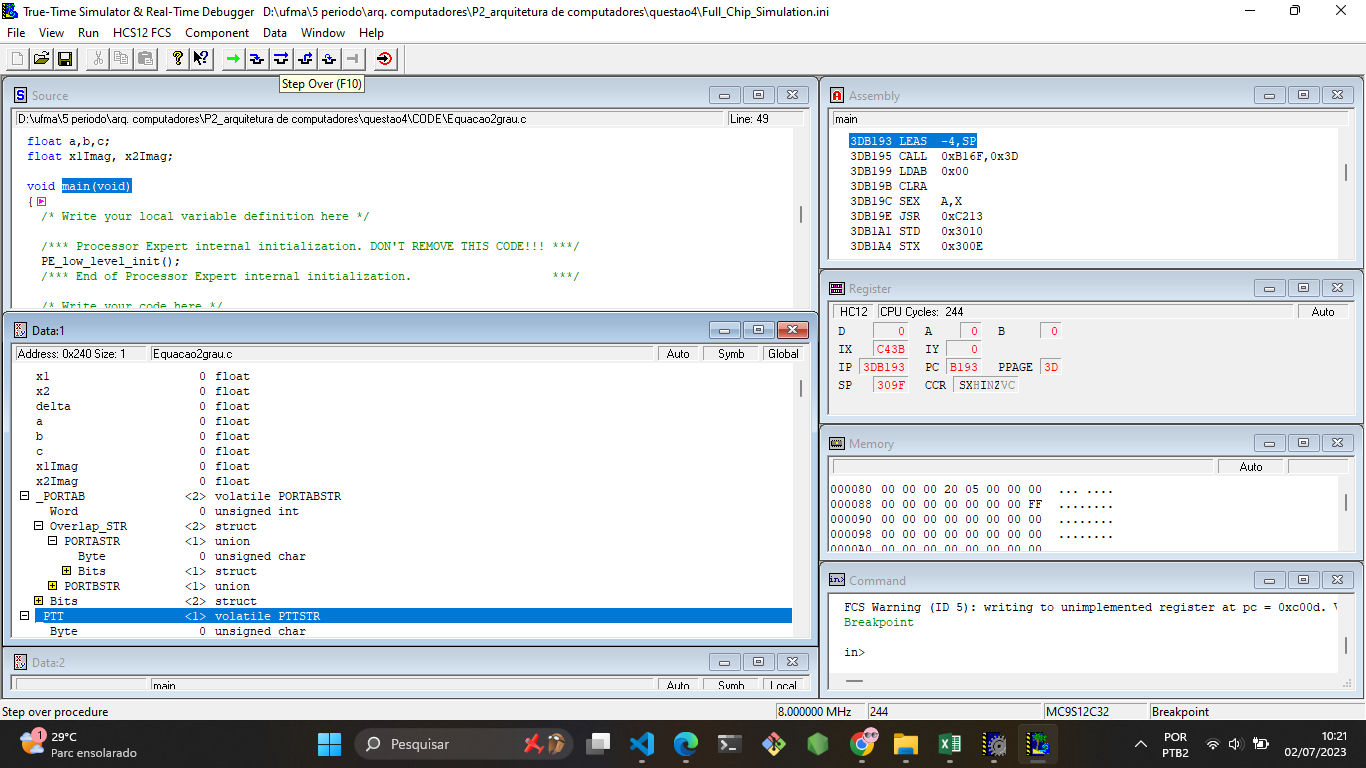
\*\* for the Freescale HCS12 series of microcontrollers.

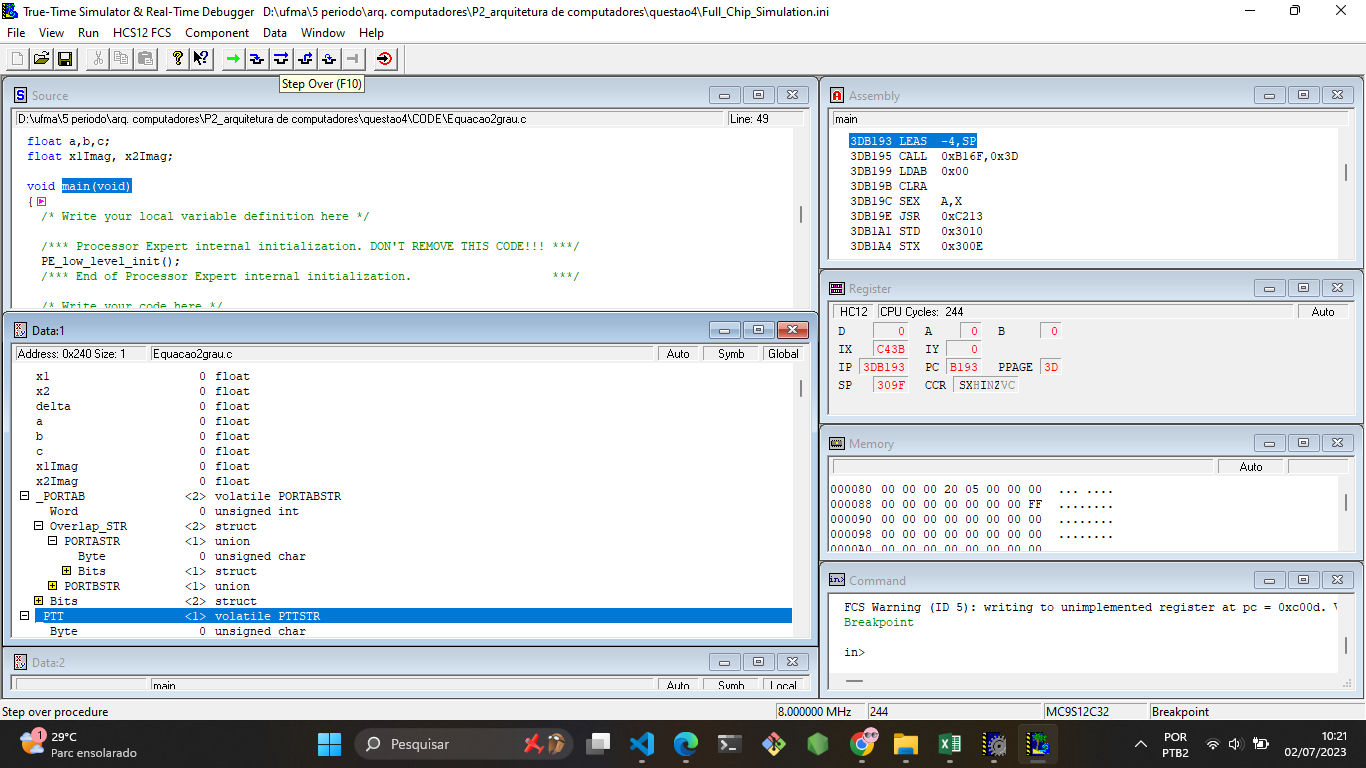
\*\*

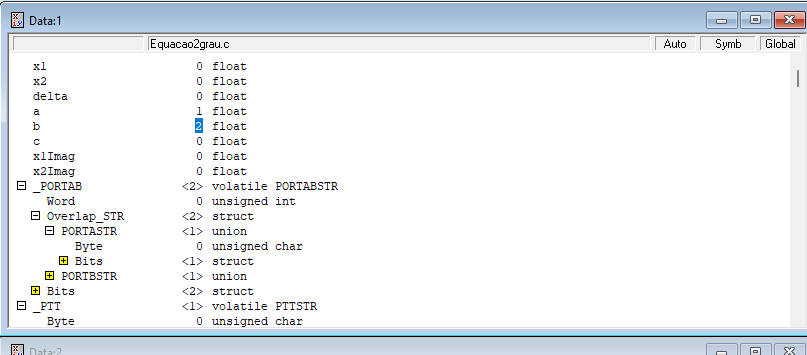
\*\* ###################################################################

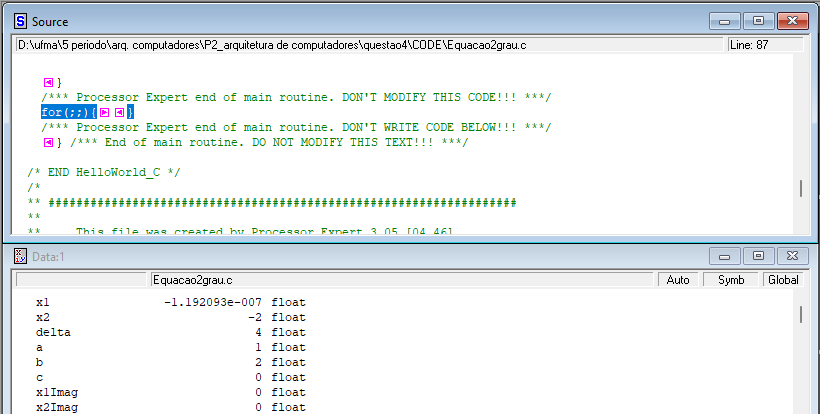
\*/











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



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 |

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

\*\* Filename : tanque.c

\*\* Project : tanque

\*\* Processor : MC9S12C128CFU16

\*\* Version : Driver 01.14

\*\* Compiler : CodeWarrior HC12 C Compiler

\*\* Date/Time : 19/04/2023, 14:13

\*\* Abstract :

\*\* Main module.

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

\*\* Settings :

\*\* Contents :

\*\* No public methods

\*\*

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

/\* MODULE tanque \*/

/\*

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

\*/

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

#include "Cpu.h"

#include "Events.h"

#include "I.h"

#include "A.h"

#include "B.h"

#include "C.h"

#include "D.h"

#include "E.h"

#include "Erro.h"

#include "S.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) \*/

int valueE;

int valueS;

int error;

int SYSTEM;

void main(void)

{

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

int valueSensorA;

int valueSensorB;

int valueSensorC;

int valueSensorD;

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

PE\_low\_level\_init();

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

/\* Write your code here \*/

while(1)

{

SYSTEM = I\_GetVal();

valueSensorA = A\_GetVal();

valueSensorB = B\_GetVal();

valueSensorC = C\_GetVal();

valueSensorD = D\_GetVal();

if(!SYSTEM){

if ((((!valueSensorA && !valueSensorB && !valueSensorD) || (!valueSensorA && valueSensorC && !valueSensorD))) != 0)

{

E\_ClrVal();

}

else

{

E\_SetVal();

}

if ((valueSensorB && valueSensorC && !valueSensorD) != 0)

{

S\_ClrVal();

}

else

{

S\_SetVal();

}

if ((valueSensorD || (valueSensorB && !valueSensorC) || (valueSensorA && !valueSensorB)) != 0)

{

Erro\_ClrVal();

}

else

{

Erro\_SetVal();

}

}

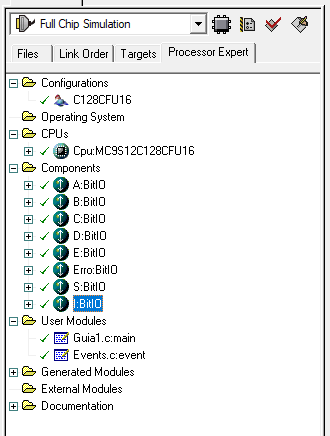
}

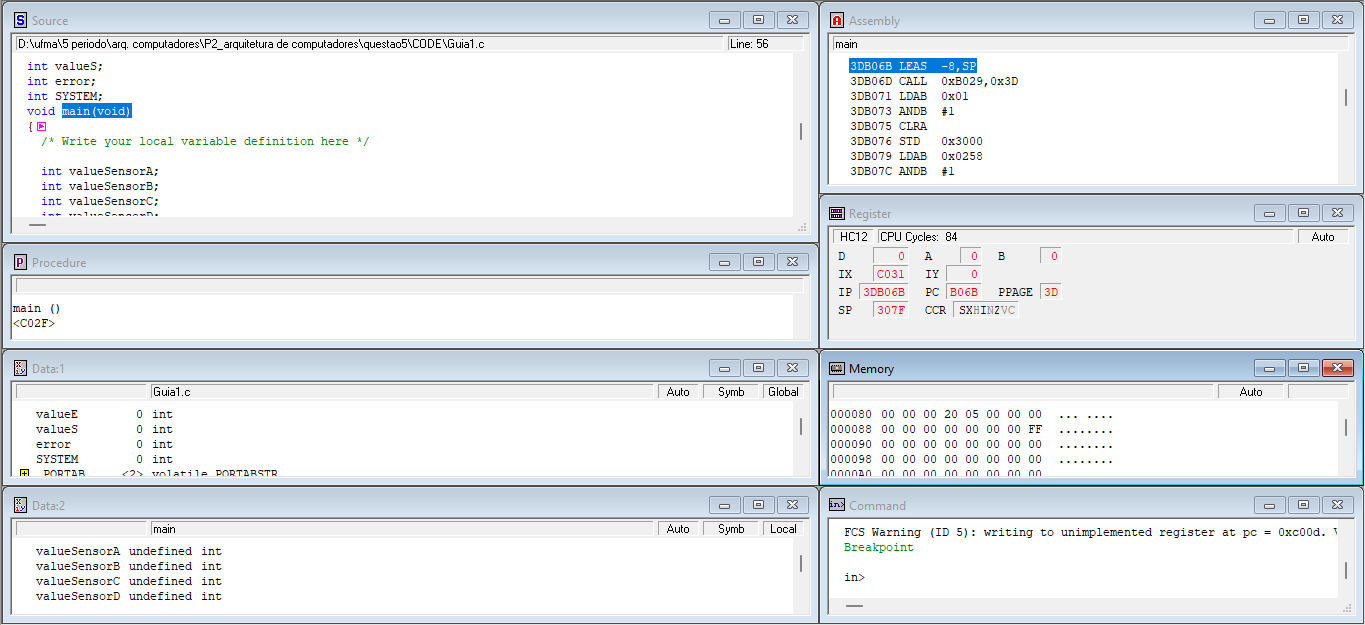
/\*\*\* Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! \*\*\*/

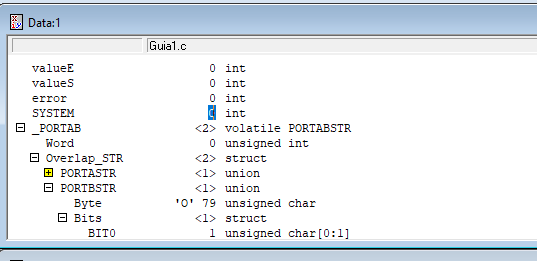
for(;;){}

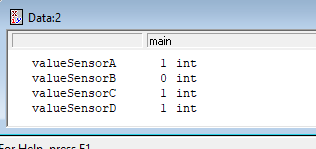
/\*\*\* Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! \*\*\*/

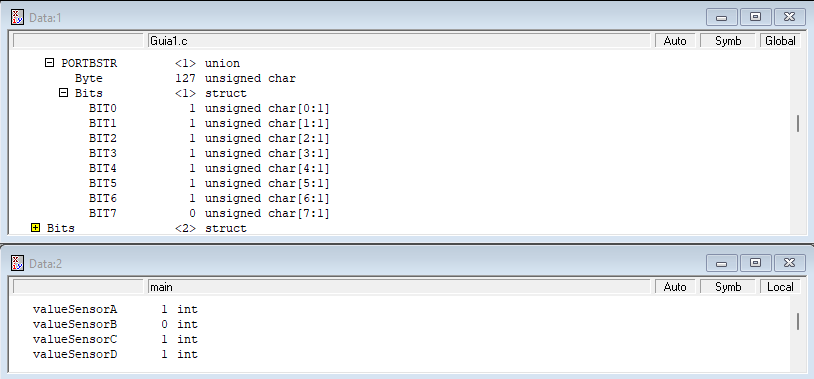
}











# 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:
2. 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:
3. 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: 02 de julho de 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.

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

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

[6] Valvano, J.2012. Embedded Systems: Introduction to ARM Cortex-M Microcontrollers. Local de publicação: Editora.

[7]Rafiquzzaman, M. (Ano). Microcontroller Theory and Applications with the PIC18F. Local de publicação: Editora.

[8] Microchip Technology. Disponível em: <www.microchip.com>. Acesso em: 02/07/2023.

[9] Embedded.com. Disponível em: <www.embedded.com>. Acesso em: 02/07/2023.