Arquitectura de Computadores

2014/2015

Máquina de Bebidas
AC_80C51

Docentes:

Dionísio Barros

Juan Jardim

Élvio Jesus

Tiago Meireles

Fábio Mendonça

Discentes:

André Braga nº 2077508

Ricardo Pereira nº 204008

Índice

3……Introdução

4…..…….Implementação e Limitações

4..……...……..Funcionamento do Programa

4……..…….Display

5……..…….Devoluções De Moedas

5…….…….…….Variáveis do Tipo sbit

6…….……..Conclusão

7…….……..Anexo A (Fluxogramas)

17…….Anexo B (Linguagem Assembly)

45…….Anexo C (Linguagem C

1. Introdução

No âmbito da cadeira de Arquitectura de Computadores, foi-nos proposto a implementação de um programa (em assembly e C) que controla-se o funcionamento de uma máquina de vendas de bebidas, recorrendo ao microcontrolador AT89S51.

A máquina de bebidas deve permitir ao utilizador introduzir dinheiro através da interface composta por botões, tendo a possibilidade de Cancelar a compra e reaver o dinheiro, ou então se o dinheiro inserido for suficiente, escolher a bebida desejada e receber a mesma.

Neste trabalho prático, o sistema utiliza dois microcontroladores AT89S51, um *Master* e um S*lave*. O *Master* é responsável pelo controlo do sistema, este comunica com o *Slave*, através da porta serial, que envia para o display o valor do total do dinheiro introduzido pelo utilizador. Os pinos do controlador *Master* estão ligados a diversos botões, motores e sensores, configurados da seguinte forma:

P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
-	-	-	-	-	Sensor Moedas 0,10€	Sensor Moedas 0,50€	Sensor Moedas 0,10€

Tabela 1 – Relação entre os pinos e os Sensores dos motores das moedas

P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
Escolher Bebida4	Escolher Bebida3	Escolher Bebida2	Escolher Bebida1	Inserir 0,50 €	Inserir 0,20 €	Inserir 0,10 €	Inserir 0,05 €

Tabela 2 – Relação entre os pinos e os botões de introdução de moedas e escolha de bebidas

P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
-	Motor Moedas 0,05€	Motor Moedas 0,50€	Motor Moedas 0,10€	Sensor Bebida4	Sensor Bebida3	Sensor Bebida2	Sensor Bebida1

Tabela 3 – Relação entre os pinos e os motores das moedas/Sensores dos motores das bebidas

P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0
Motor Bebida4	Motor Bebida3	Motor Bebida2	Motor Bebida1	-	Botão Cancel	Porta Serial	-

Tabela 4 – Relação entre os pinos e Porta Serial, Cancelar e motores das bebidas

2. Implementação e Limitações

Nesta secção serão abordados alguns aspectos importantes da implementação, que ajudam a clarificar o funcionamento do programa assim como algumas limitações, tanto de hardware como de software.

2.1. Funcionamento do programa

Seguindo a sugestão do professor Élvio, decidimos implementar o programa com estados.

O estado 1 é dedicado à introdução de moedas e selecção da bebida escolhida, neste estado é lido e acumulado o valor das moedas introduzidas, esse mesmo valor é enviado para o display. O estado 1 também permite escolher a bebida desejada (se existir em stock) e caso o dinheiro introduzido seja suficiente para a compra da bebida, o programa passa para o estado 2. De salientar também que durante a execução deste estado o utilizador pode carregar no botão Cancel (interrupção externa) para reaver o dinheiro introduzido, caso isto aconteça o programa muda para o estado 3.

O estado 2 é o estado de "venda", durante a sua execução é calculado o troco a dar ao utilizador, sendo este devolvido de seguida, caso seja necessário. Logo de seguida é dada a bebida escolhida ao utilizador. Pode ocorrer um caso em que a máquina não tem moedas em stock para devolver o troco, neste caso a máquina dá apenas a bebida escolhida ao utilizador. Quando este estado termina, o programa volta a executar o estado 1.

O estado 3 é executado quando o utilizador carrega no botão Cancel enquanto o programa está no 1º estado. Neste estado é devolvido ao utilizador o dinheiro total inserido até ao momento. Depois de executar este estado, o programa volta para o estado 1.

2.2. Limitações

2.2.1.Display

Visto que o display só consegue apresentar valores até 255, qualquer valor de moedas introduzido que ultrapasse este valor faz com que a máquina funcione incorrectamente.

2.2.2.Devolução de moedas

É sabido também que a máquina quando tem de devolver cinco moedas de 50 cêntimos, em casos raros, devolve uma moeda a menos.

Um caso muito mais frequente ocorre na devolução de moedas de 10 cêntimos, em que a máquina retorna apenas metade, ou menos de metade, da quantidade de moedas que devia devolver. Estas situações nunca ocorrem quando o software é simulado num computador.

2.2.3.Variáveis do tipo *sbit*

Na implementação, em linguagem C, tentamos criar uma função genérica para rodar qualquer um dos motores. Esta função recebia como parâmetros um determinado motor e o respectivo sensor, ambas variáveis do tipo *sbit*. Depois de alguma pesquisa descobrimos que não é possível utilizar este tipo de variáveis como parâmetros de funções, o que impossibilitou a utilização de uma função genérica para rodar os motores, logo tivemos de criar uma função individual para rodar cada motor.

Esta função encontra-se “comentada” no fim do código em linguagem C.

3. Conclusão

Em vários aspectos, a realização deste trabalho foi muito interessante, sendo que nos proporcionou trabalhar pela primeira vez com um microcontrolador.

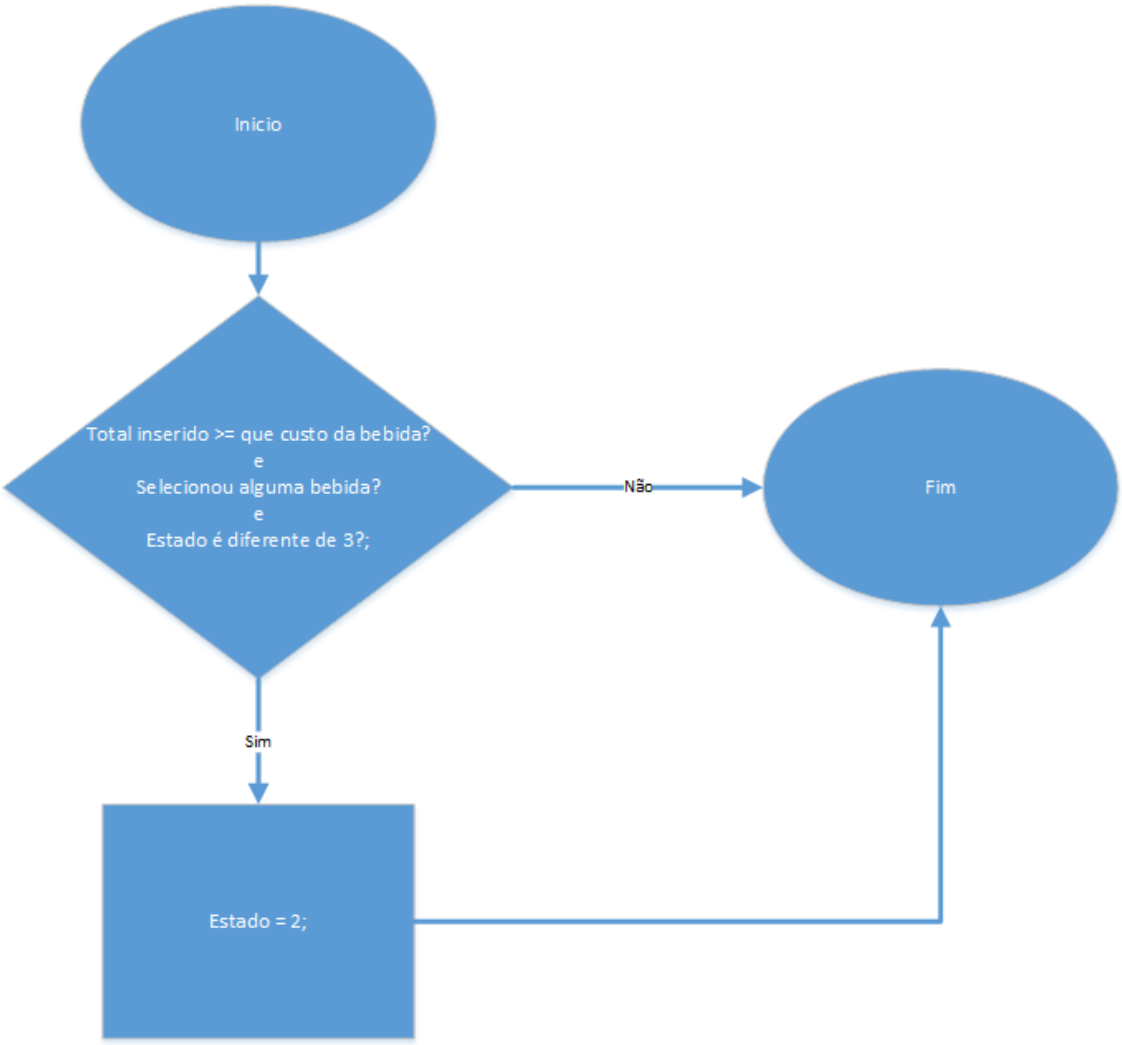
Após perceber o funcionamento do microcontrolador, desenvolver o código em linguagem C foi relativamente fácil, pois já tínhamos experiencia na utilização desta mesma linguagem.

Relativamente à programação em assembly, em confronto com o PEPE, este microcontrolador é mais simples de programar, pois permite acesso directo à memoria e a possibilidade de fazer comparações e saltos com uma só instrução. Devido a estas razões, o desenvolvimento do código em assembly tornou-se praticamente uma tradução directa do código em linguagem C. Apercebemo-nos que, apesar do PEPE e o AT89S51 funcionarem de forma diferente e terem instruções diferentes, a lógica de programação é extremamente semelhante.

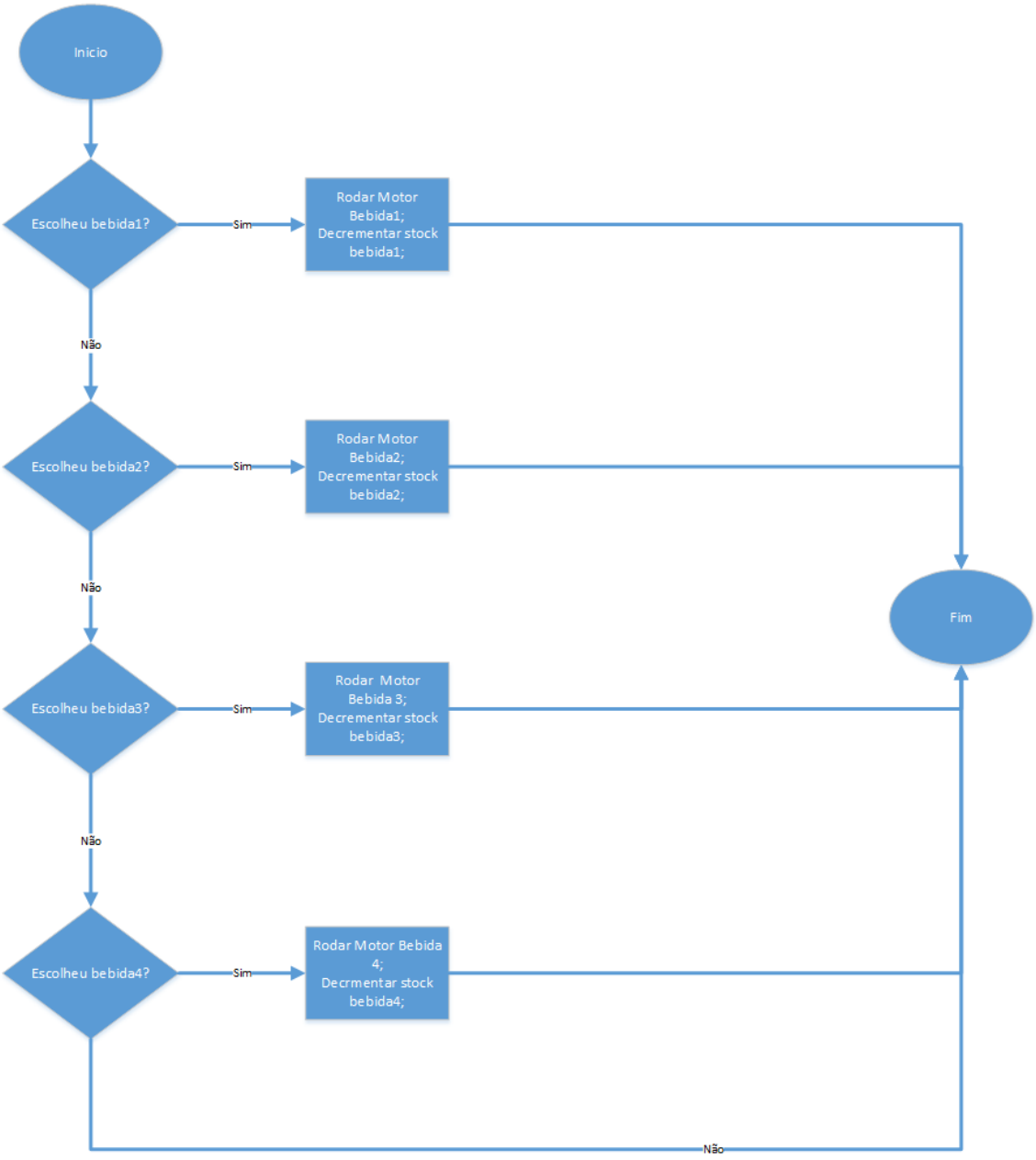
A grande dificuldade na realização deste projecto residiu na máquina de bebidas em si. Esta não devolvia o troco correctamente, como já referimos anteriormente, e os motores da mesma, nomeadamente o da bebida2, o das moedas de 0,10 € e o das moedas de 0,05€ deixaram de funcionar.

A realização deste projecto permitiu-nos aplicar todo o conhecimento adquirido ao longo do semestre, o que nos ajudou a consolidar a formação recebida. Este é um património conseguido para o futuro.

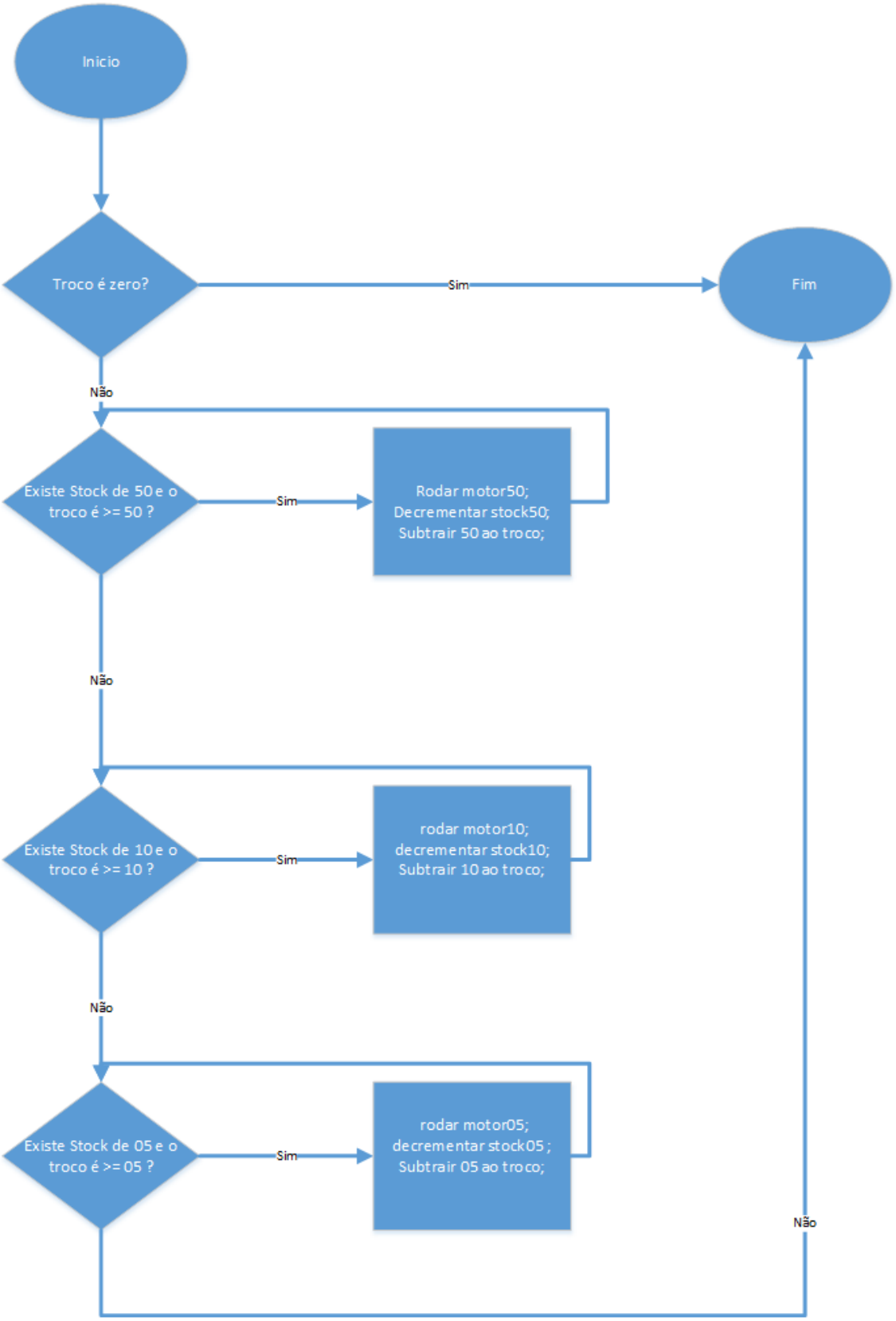
4. Anexo A - Fluxogramas



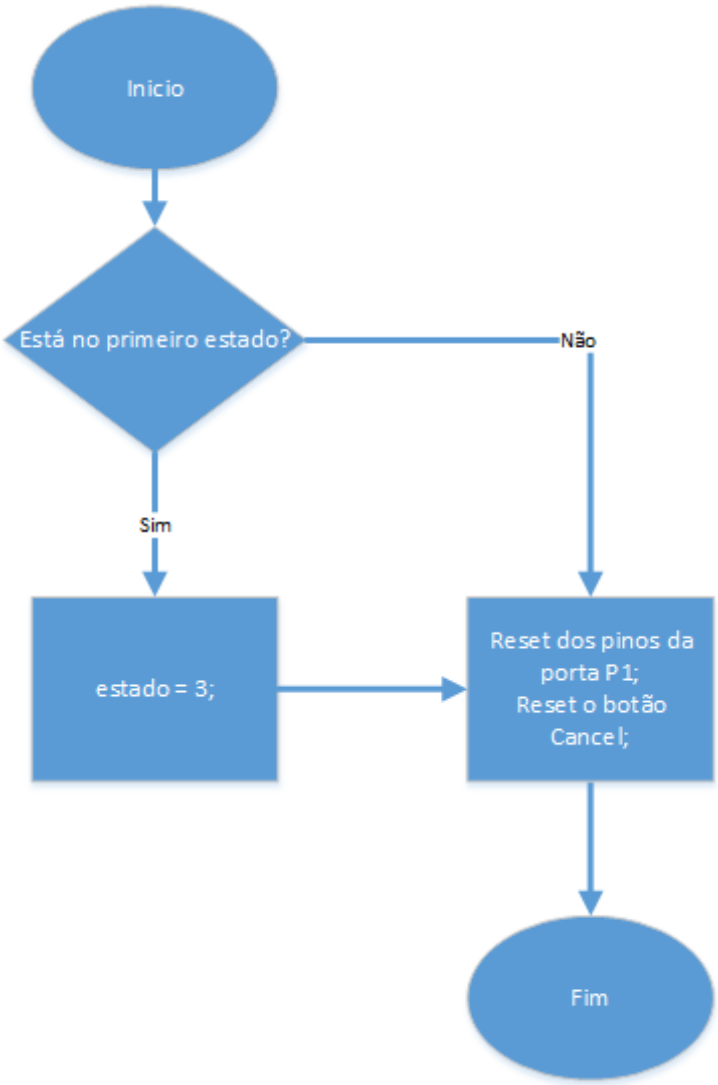
Fluxograma 1 – checkTotalCusto



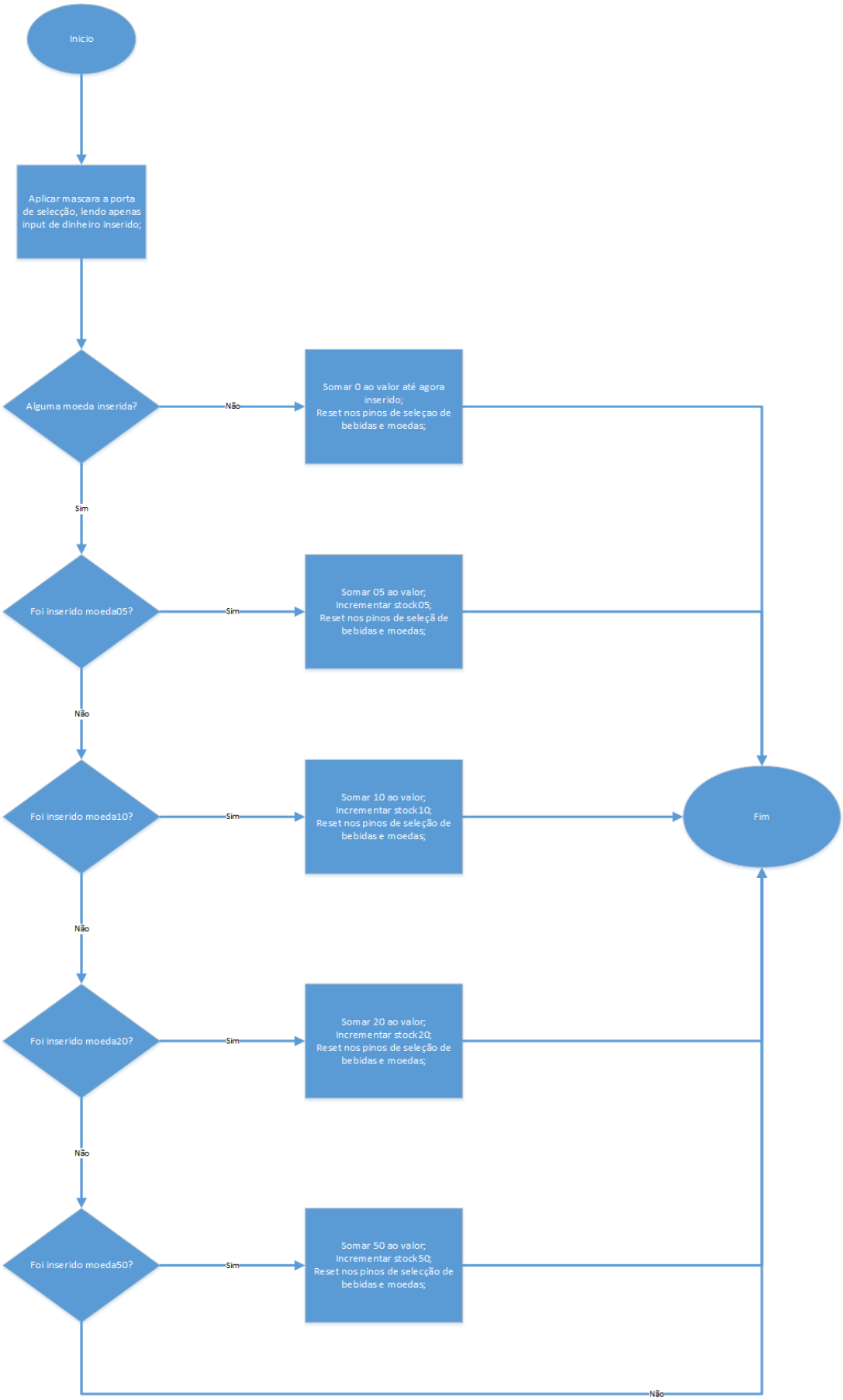
Fluxograma 2 - darBebida



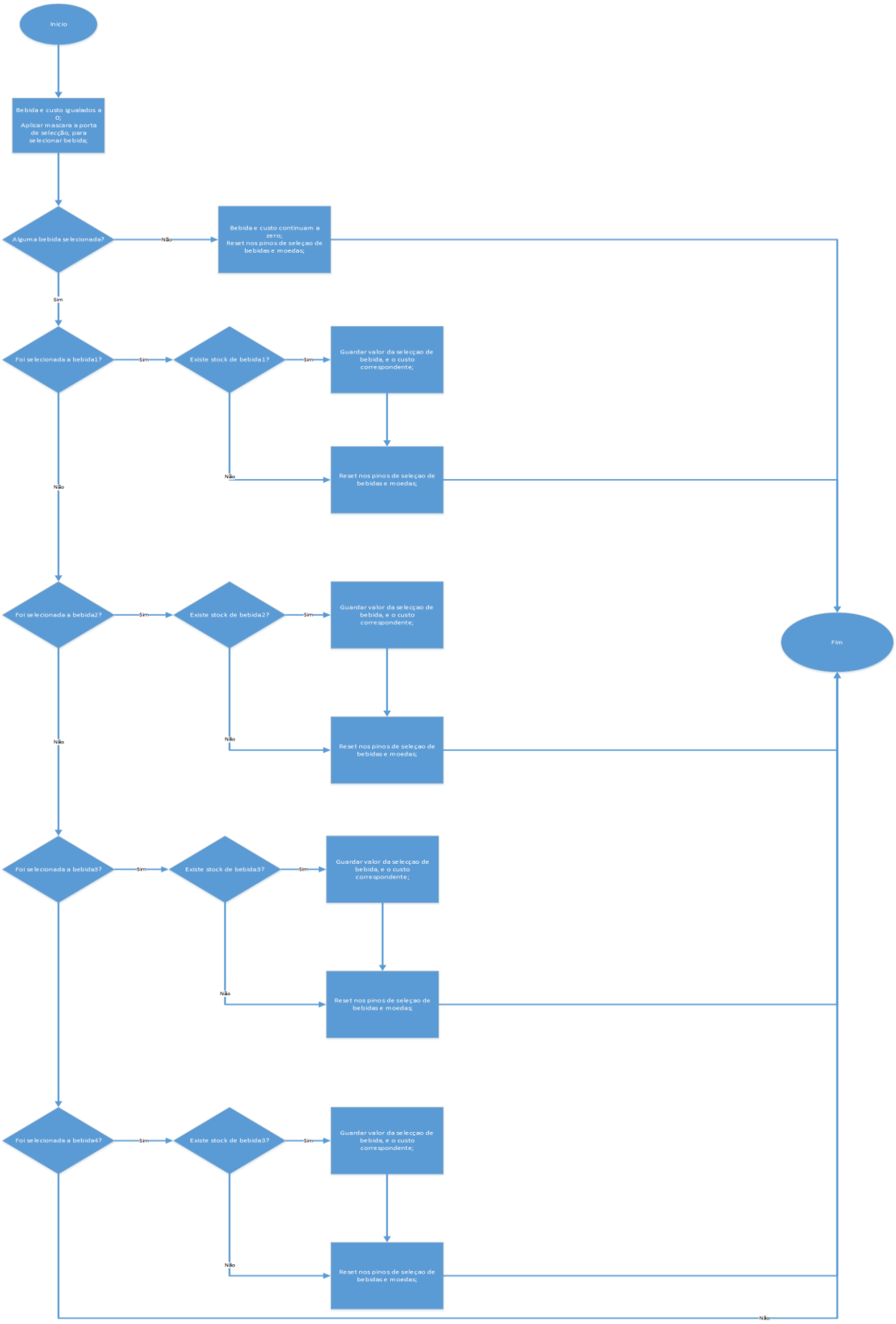
Fluxograma 3 – devolverTroco



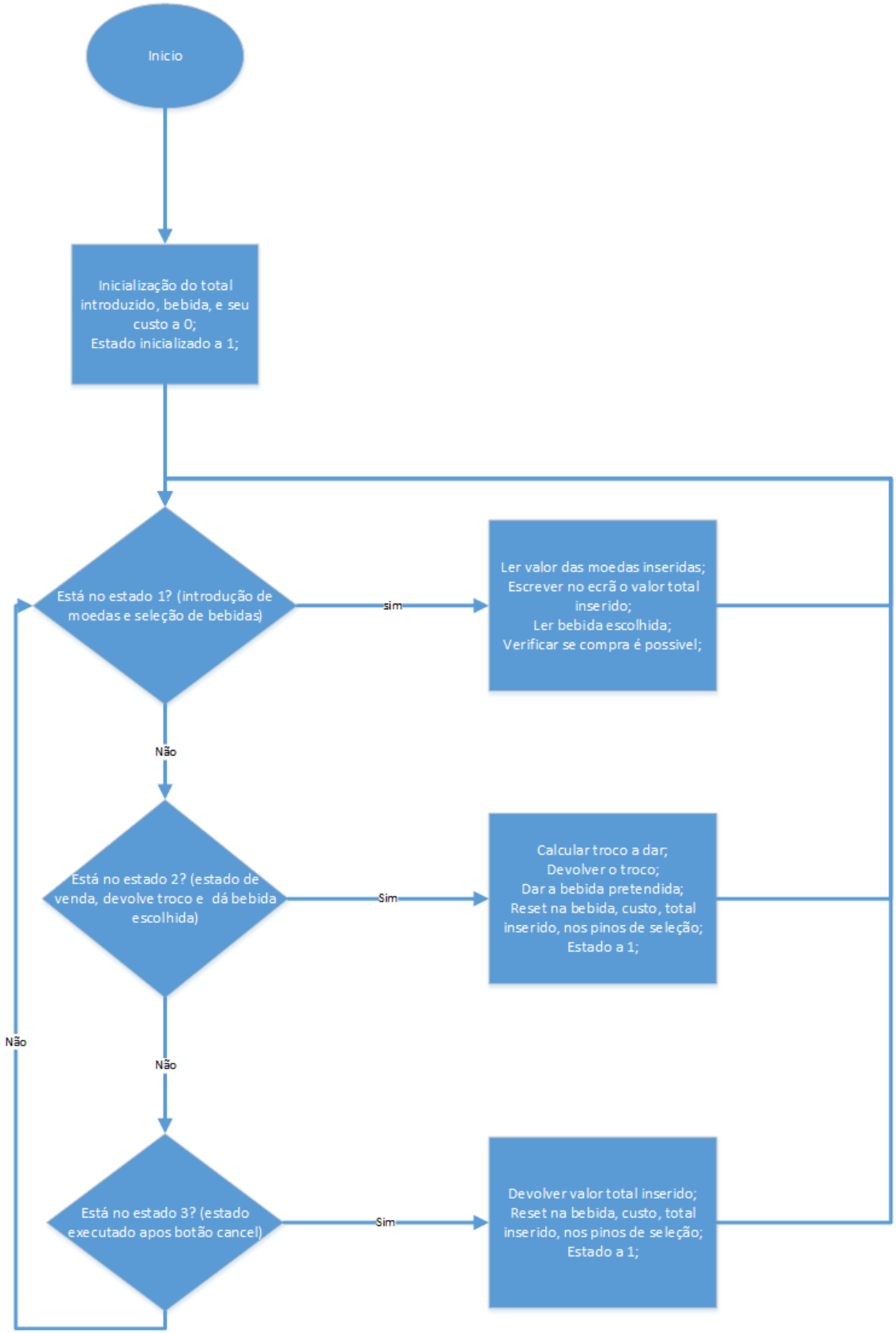
Fluxograma 4 - external_int0



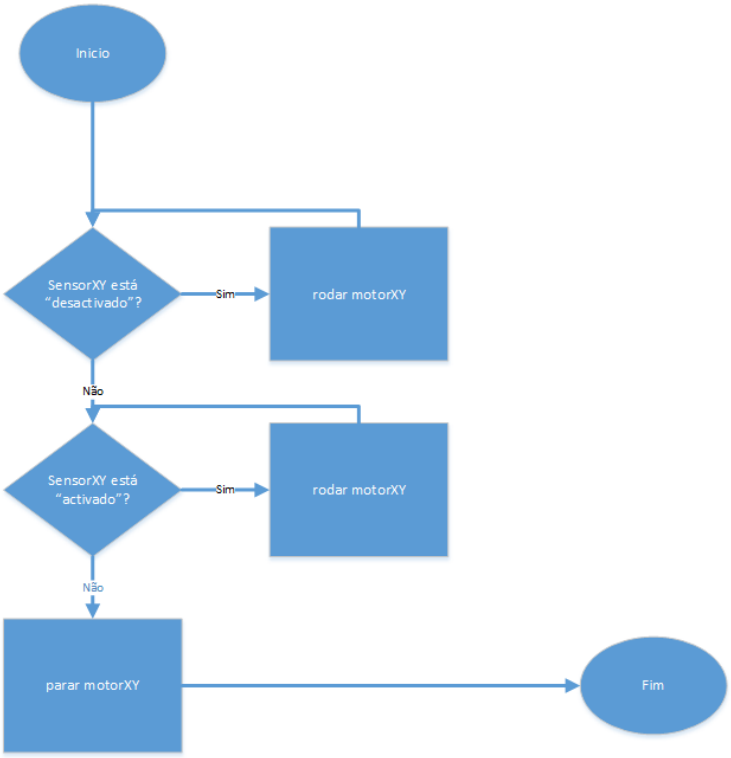
Fluxograma 5 – ler_Valor



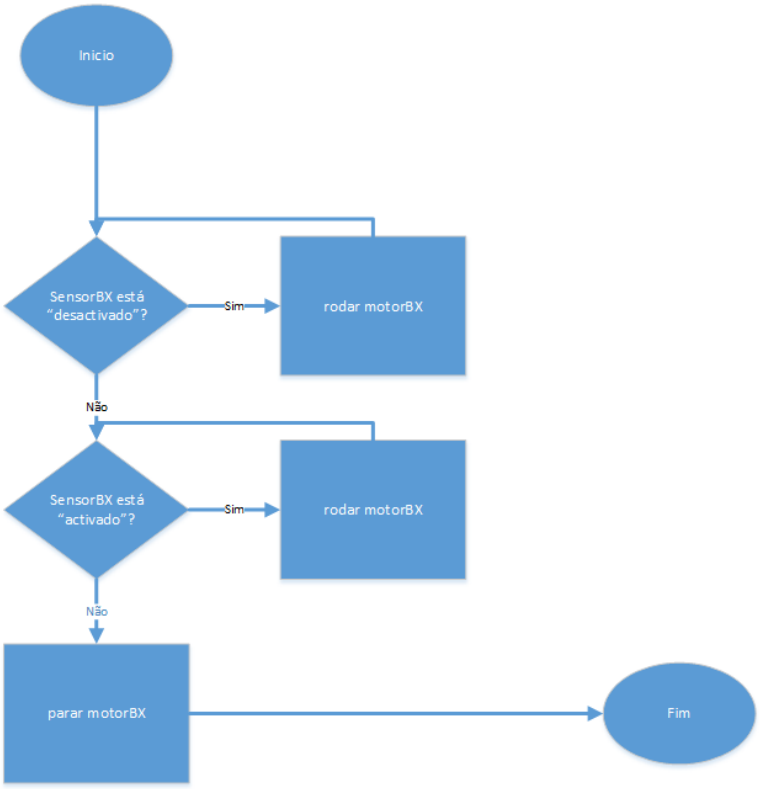
Fluxograma 6 – lerBebida



Fluxograma 7 – Programa principal

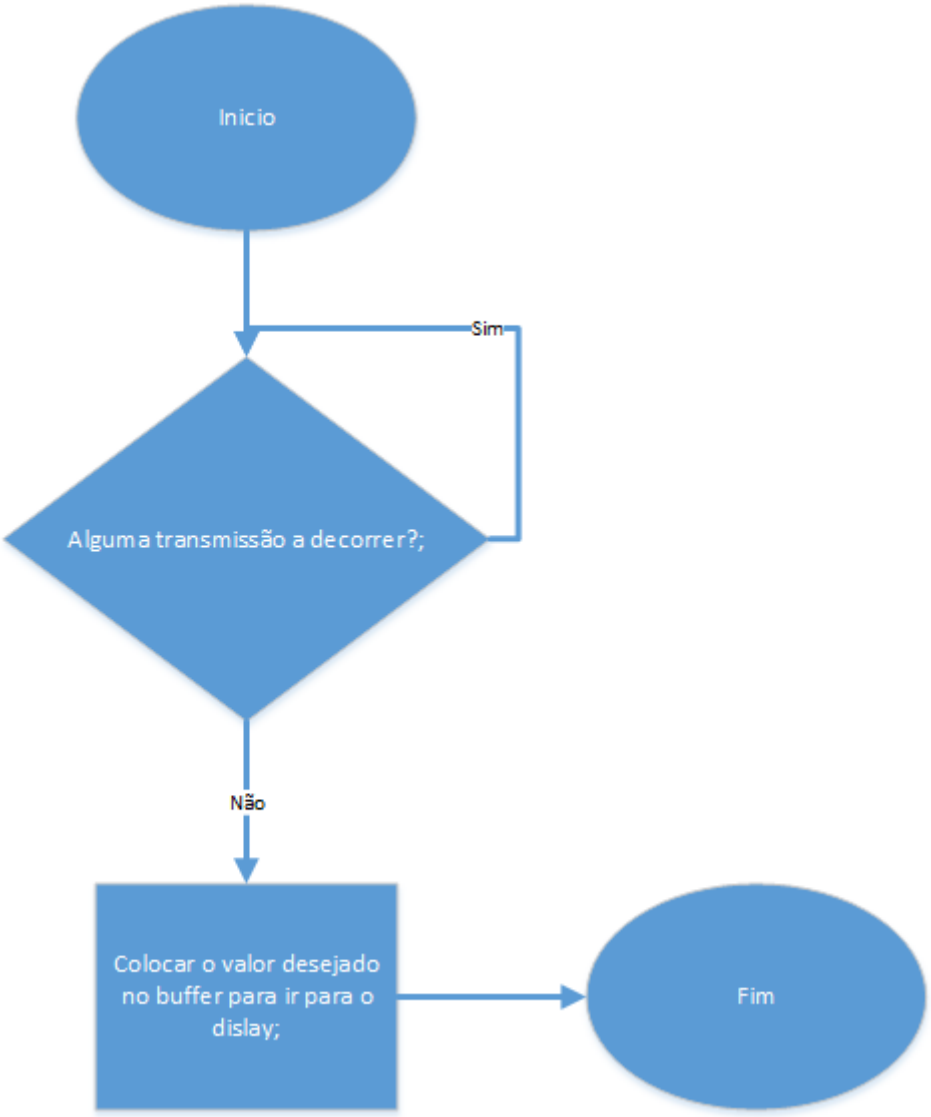


Fluxograma 8 – rodar Motor(moedas)



Fluxograma 9 - rodarMotorB(bebidas)

Fluxograma 10 - calcularTroco



Fluxograma 11 – TXD_data

5. Anexo B – Código em linguagem Assembly

```
;------------------------------;
; Definicao das Portas e Pinos ;
;------------------------------;

Pselecao EQU P1          ;Porta dos botoes de seleção das moedas e bebidas

; Pinos dos Motores das bebidas
MotorB1 EQU P3.4 ;Pino do motor da bebida 1
MotorB2 EQU P3.5 ;Pino do motor da bebida 2
MotorB3 EQU P3.6 ;Pino do motor da bebida 3
MotorB4 EQU P3.7 ;Pino do motor da bebida 4

; Pinos dos Sensores dos motores das bebidas
SensorB1 EQU P2.0 ;Pino do sensor do motor da bebida 1
SensorB2 EQU P2.1 ;Pino do sensor do motor da bebida 2
SensorB3 EQU P2.2 ;Pino do sensor do motor da bebida 3
SensorB4 EQU P2.3 ;Pino do sensor do motor da bebida 4

; Pinos dos Motores das moedas
MotorM05 EQU P2.6 ;Pino do motor da moeda de 0,05 euros
MotorM10 EQU P2.4 ;Pino do motor da moeda de 0,10 euros
MotorM50 EQU P2.5 ;Pino do motor da moeda de 0,50 euros

; Pinos dos Sensores dos motores das moedas
SensorM05 EQU P0.2 ;Pino do sensor do motor da moeda de 0,05 euros
SensorM10 EQU P0.0 ;Pino do sensor do motor da moeda de 0,10 euros
SensorM50 EQU P0.1 ;Pino do sensor do motor da moeda de 0,50 euros
```

```
; Pinos do cancel e porta serial
Cancel    EQU P3.2 ;Pino do botão Cancel
Pserial EQU P3.1 ;Pino da porta serial

;----------;
; variaveis ;
;----------;

; stock das moedas existentes ;
StockM05 EQU 0x78 ;stock de moedas de 0,05
StockM10 EQU 0x79 ;stock de moedas de 0,10
StockM20 EQU 0x7A ;stock de moedas de 0,20
StockM50 EQU 0x7B ;stock de moedas de 0,50

; stock das bebidas existentes ;
StockB1 EQU 0x7C ;Stock da bebida1
StockB2 EQU 0x7D ;Stock da bebida2
StockB3 EQU 0x7E ;Stock da bebida3
StockB4 EQU 0x7F ;Stock da bebida4

;--------------;
;  constantes  ;
;--------------;

; custo das bebidas existentes ;
CustoB1 EQU 80  ;Custo da bebida1
CustoB2 EQU 100 ;Custo da bebida2
CustoB3 EQU 120 ;Custo da bebida3
CustoB4 EQU 150 ;Custo da bebida4
```

```
CSEG AT 0000H
        JMP Inicio

;se ocorrer interrupçao externa 0, salta para este endereço
CSEG AT 0003H
        JMP InterrupcaoExt0     ;Saltar para a rotina de tratamento da interrupção externa0

CSEG AT 0050H
Inicio:

        MOV SP, #7 ;inicializa stack pointer no endereço 07H

        CALL InicializarStocks              ;inicializa stocks de bebidas e moedas
        CALL InicializarPortas              ;inicializa valores das portas
        CALL ActivarTransPortaSerial    ;activar transmissao da porta serial
        CALL Interrupcoes_init              ;inicializar interrupções
        CALL CalibrarMotores                ;calibra os motores das bebidas e das moedas

        ;R0 variavel de ESTADO que determina em que estado esta o programa
        MOV R0, #1          ;Programa começa no estado 1

        ;R1 variavel com o TOTAL de dinheiro introduzido em centimos
        MOV R1, #0          ;valor total introduzido começa em 0

        ;R2 variavel com o CUSTO da bebida escolhida
        MOV R2, #0          ;custo da bebida é 0 no inicio

        ;R3 variavel que indica qual a BEBIDA selecionada
```

```
        MOV R3, #0              ;no inicio não há nenhuma bebida escolhida

        ;R4 variavel com o TROCO a dar
        MOV R4, #0              ;no inicio não há troco a dar



;------------------;
;  Ciclo Infinito  ;
;------------------;
;R0 contem o valor do ESTADO
;R1 contem o valor TOTAL de dinheiro introduzido
;R2 contem o CUSTO da bebida escolhida
;R3 contem o numero da BEBIDA escolhida
;R4 contem o TROCO a dar

CICLO_PRINCIPAL:


;introdução de moedas e seleção de bebida
CASO1:

        CJNE R0, #1, CASO2      ;Se valor do estado(R0) nao for 1, salta para o caso2
        CALL Ler_valor          ;lê e acumula valor das moedas introduzidas
        CALL TXD_data                   ;escreve no display o valor total introduzido
        CALL Ler_bebida                 ;lê a bebida escolhida e guarda o custo da mesma
    CALL CheckTotalCusto ;Verifica se o valor total de moedas introduzidas é maior que o custo da bebida escolhida
        JMP CICLO_PRINCIPAL     ;Salta para o inicio do ciclo


;devolve troco e dá a bebida escolhida
CASO2:
```

```
        CJNE R0, #2, CASO3      ;Se valor do estado(R0) nao for 2, salta para o caso3
        CALL CalcularTroco      ;calcula o troco a dar
        CALL DevolverTroco  ;devolve o troco a dar
        CALL DarBebida              ;dá a bebida escolhida
        MOV R4, #0                  ;reset do TROCO a dar
        MOV R3, #0                  ;reset da BEBIDA escolhida
        MOV R2, #0                  ;reset do CUSTO da bebida escolhida
        MOV R1, #0                  ;reset do TOTAL de dinheiro inserido
        MOV R0, #1                  ;Estado volta a ser 1
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        JMP CICLO_PRINCIPAL         ;Salta para o inicio do ciclo

;executado após o botão cancel ser pressionado
CASO3:

        CJNE R0, #3, CICLO_PRINCIPAL        ;Se valor do estado(R0) nao for 3, salta para o inicio
do ciclo
        MOV A, R1
        MOV R4, A                   ;TROCO a dar (R4) passa a ter o valor do TOTAL de
dinheiro introduzido (R1)
        CALL DevolverTroco  ;devolve o dinheiro total introduzido
        MOV R4, #0                  ;reset do TROCO a dar
        MOV R3, #0                  ;reset da BEBIDA escolhida
        MOV R2, #0                  ;reset do CUSTO da bebida escolhida
        MOV R1, #0                  ;reset do TOTAL de dinheiro inserido
        MOV R0, #1                  ;Estado volta a ser 1
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        JMP CICLO_PRINCIPAL         ;Salta para o inicio do ciclo

;---------FIM DO CICLO_PRINCIPAL----------
```

```
;----------------------;
;   ROTINA LER_VALOR   ;
;----------------------;
;le os valores das moedas inseridas, adicionando-as ao total

Ler_valor:

;R1 contem o valor TOTAL de moedas introduzidas

        MOV A, #0x0F   ;introducao da mascara, guardada no acumulador
        ANL A, Pselecao ;executa uma operaçao AND bit a bit da mascara(que esta em A) com a porta de seleção de moedas e bebidas
                                        ;passando para 0 os valores dos pinos referentes às bebidas, guarda resultado no acumulador

        MOV R5, A               ;Guarda resultado da operação anterior em R5, R5 contem valores dos pinos referentes às moedas (os das bebidas estao a 0)
                                        ;a combinação de valores em R5 determina qual das moedas foi selecionada, esta combinação está em decimal

        MOV A, R1               ;move para o acumulador o valor TOTAL de moedas introduzidas (R1)

CASO15:

        CJNE R5, #15, CASO14 ;Verficar sem em R5 tem o valor 15 (0000 1111b), se não salta para o CASO14
        ADD A, #0                       ;Não foram introduzidas moedas, somamos 0
        MOV R1, A                       ;Actualizar TOTAL de moedas introduzidas (R1)
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        JMP FIM_LER_VALOR   ;Salta para o fim da rotina
```

```
CASO14:

        CJNE R5, #14, CASO13 ;Verficar sem em R5 tem o valor 14 (0000 1110b), se não salta para o
CASO13
        ADD A, #5                   ;Introduzida moeda 5 centimos, somamos 5
        MOV R1, A                   ;Actualizar TOTAL de moedas introduzidas (R1)
        INC StockM05            ;Adiciona uma moeda ao stock de 5 centimos
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
        JMP FIM_LER_VALOR

CASO13:

        CJNE R5, #13, CASO11 ;Verficar sem em R5 tem o valor 13 (0000 1101b), se não salta para o
CASO11
        ADD A, #10                  ;Introduzida moeda 10 centimos, somamos 10
        MOV R1, A                   ;Actualizar TOTAL de moedas introduzidas (R1)
        INC StockM10            ;Adiciona uma moeda ao stock de 10 centimos
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
        JMP FIM_LER_VALOR  ;Salta para o fim da rotina

CASO11:

        CJNE R5, #11, CASO7 ;Verficar sem em R5 tem o valor 11 (0000 1011b), se não salta para o
CASO7
        ADD A, #20                  ;Introduzida moeda 20 centimos, somamos 20
        MOV R1, A                   ;Actualizar TOTAL de moedas introduzidas (R1)
        INC StockM20            ;Adiciona uma moeda ao stock de 20 centimos
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
```

```
        JMP FIM_LER_VALOR  ;Salta para o fim da rotina

CASO7:

        CJNE R5, #7, FIM_LER_VALOR ;;Verficar sem em R5 tem o valor 7 (0000 0111b), se não
salta para o fim da rotina
        ADD A, #50                  ;Introduzida moeda 50 centimos, somamos 50
        MOV R1, A                   ;Actualizar TOTAL de moedas introduzidas (R1)
        INC StockM50            ;Adiciona uma moeda ao stock de 50 centimos
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
        JMP FIM_LER_VALOR  ;Salta para o fim da rotina

FIM_LER_VALOR:

        RET

;------FIM ROTINA LER_VALOR-------



;-----------------------;
;   ROTINA LER_BEBIDA   ;
;-----------------------;
;le bebida escolhida e guarda o custo da mesma

Ler_bebida:

;R2 contem o CUSTO da bebida
;R3 contem a BEBIDA escolhida

        MOV A, #0xF0   ;introducao da mascara, guardada no acumulador
```

```
        ANL A, Pselecao ;executa uma operaçao AND bit a bit da mascara(que esta em A) com a porta de seleção de moedas e bebidas
                                                ;passando para 0 os valores dos pinos referentes às moedas, guarda resultado no acumulador

        MOV R5, A               ;Guarda resultado da operação anterior em R5, R5 contem valores dos pinos referentes às bebidas (os das moedas estao a 0)
                                                ;a combinação de valores em R5 determina qual das bebidas foi selecionada, esta combinação está em decimal

        MOV R2, #0              ;como ainda não foi escolhida nenhuma bebida o CUSTO é zero
        MOV R3, #0              ;BEBIDA escolhida é zero (pois ainda não foi escolhida nenhuma)

CASO240:

        CJNE R5, #240, CASO224  ;Verficar sem em R5 tem o valor 240 (1111 0000b), se não salta para o CASO224
        MOV R3, #0                      ;Não foi selecionada nenhuma bebida
        MOV R2, #0                      ;CUSTO é zero
        MOV Pselecao, #0xFF     ;Reset dos pinos de seleção de moedas e bebidas
        JMP FIM_LER_BEBIDA      ;Salta para o fim da rotina

CASO224:

        CJNE R5, #224, CASO208  ;Verficar sem em R5 tem o valor 224 (1110 0000b), se não salta para o CASO208
        MOV A, StockB1                  ;Mover Stock da bebida1 para o acumulador (JZ verifica o valor de A)
        JZ noStockB1            ;Se o Stock da bebida1 for zero, as duas proximas instruçoes não são executadas

        MOV R3, #1                      ;Selecionada bebida1
        MOV R2, #CustoB1        ;CUSTO da bebida escolhida passa a ter o valor do custo da bebida1
```

```
        noStockB1:
        MOV Pselecao, #0xFF         ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
        JMP FIM_LER_BEBIDA          ;Salta para o fim da rotina

CASO208:

        CJNE R5, #208, CASO176      ;Verficar sem em R5 tem o valor 208 (1101 0000b), se não
salta para o CASO176
        MOV A, StockB2                  ;Mover Stock da bebida2 para o acumulador (JZ
verifica o valor de A)
        JZ noStockB2                ;Se o Stock da bebida2 for zero, as duas proximas instruçoes
não são executadas

        MOV R3, #2                      ;Selecionada bebida2
        MOV R2, #CustoB2            ;CUSTO da bebida escolhida passa a ter o valor do custo da
bebida2

        noStockB2:
        MOV Pselecao, #0xFF         ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
        JMP FIM_LER_BEBIDA          ;Salta para o fim da rotina

CASO176:

        CJNE R5, #176, CASO112      ;Verficar sem em R5 tem o valor 176 (1011 0000b), se não
salta para o CASO112
        MOV A, StockB3                  ;Mover Stock da bebida3 para o acumulador (JZ
verifica o valor de A)
        JZ noStockB3                ;Se o Stock da bebida3 for zero, as duas proximas instruçoes
não são executadas
```

```
        MOV R3, #3                      ;Selecionada bebida3
        MOV R2, #CustoB3                ;CUSTO da bebida escolhida passa a ter o valor do custo da
bebida3

        noStockB3:
        MOV Pselecao, #0xFF             ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
        JMP FIM_LER_BEBIDA              ;Salta para o fim da rotina

CASO112:

        CJNE R5, #112, FIM_LER_BEBIDA ;Verficar sem em R5 tem o valor 112 (0111 0000b), se
não salta para o fim da rotina
        MOV A, StockB4                  ;Mover Stock da bebida4 para o acumulador (JZ
verifica o valor de A)
        JZ noStockB4                    ;Se o Stock da bebida4 for zero, as duas proximas instruçoes
não são executadas

        MOV R3, #4                      ;Selecionada bebida4
        MOV R2, #CustoB4                ;CUSTO da bebida escolhida passa a ter o valor do custo da
bebida4

        noStockB4:
        MOV Pselecao, #0xFF             ;Reset dos pinos de seleção de moedas e bebidas
        CALL mydelay
        JMP FIM_LER_BEBIDA              ;Salta para o fim da rotina


FIM_LER_BEBIDA:

        RET
```

```
;------FIM ROTINA LER_BEBIDA-------



;------------------------------;

;   ROTINA CHECK TOTAL CUSTO   ;

;------------------------------;

;Verifica se o valor total de moedas introduzidas é maior que o custo da bebida escolhida


CheckTotalCusto:

;R0 contem o valor do ESTADO

;R1 contem o valor TOTAL de moedas introduzidas

;R2 contem o CUSTO da bebida

;R3 contem a BEBIDA escolhida


        MOV A, R3                       ;move numero da BEBIDA escolhida(R3) para o acumulador (JZ verifica o valor de A)

        JZ FIM_CHECKTOTALCUSTO  ;se o valor for igual a zero (nenhuma bebida escolhida), salta para o fim da rotina

        MOV A, R1                       ;move valor TOTAL de moeda introduzidas(R1) para o acumulador

        CJNE A, 0x02, notEQR2  ;esta operacao é executada com o intuito de verificar o valor da flag de carry

                                                ;como não é permitida a comparação de dois registos directamente é usado o endereço de R2, 0x02 ou (02H)

        notEQR2:

        JC FIM_CHECKTOTALCUSTO  ;se C = 1, TOTAL < CUSTO, salta para o fim da rotina

                                                ;se C = 0, TOTAL >= CUSTO, podemos passar para o proximo estado


        MOV R0, #2                      ;muda para o ESTADO 2 para serem executadas as funçoes desse mesmo estado


FIM_CHECKTOTALCUSTO:
```

```
        RET

;------FIM ROTINA CHECK TOTAL CUSTO------



;---------------------;
;   ROTINA MY DELAY   ;
;---------------------;
;rotina que gera um atraso
;"conta" ate 255, 255 vezes (65025)
;a R6 é incrementado uma unidade sempre que R7 chega a 255
;quando R6 chega a 255 a acaba a rotina
mydelay:

        MOV R6, #0

cicloDelay1:
        MOV A, R6
        ADD A, #1
        MOV R6, A       ;incrementa uma unidade a R6

        MOV R7, #0      ;reset do registo R7

cicloDelay2:
        MOV A, R7
        ADD A, #1
        MOV R7, A       ;incrementa uma unidade a R7

        CJNE R7, #255, cicloDelay2      ;verifica se R7 chegou a 255 se não executa o ciclodelay2
```

```
        CJNE R6, #255, cicloDelay1      ;verifica se R6 chegou a 255 se não executa o ciclodelay1

        RET

;------FIM ROTINA MY DELAY------

;---------------------------;
;   ROTINA CALCULAR TROCO   ;
;---------------------------;
;calcula o troco a dar

CalcularTroco:
;R1 contem o valor TOTAL de moedas introduzidas
;R2 contem o CUSTO da bebida
;R4 TROCO a dar

        MOV A, R1       ;move valor TOTAL de moedas introduzidas (R1) para o acumulador
        CLR C           ;limpa o valor da flag de carry (pois é usado na operação SUBB)
        SUBB A, R2      ;subtrai CUSTO da bebida (R2) ao acumulador, obtendo o valor do TROCO a
dar
        MOV R4, A       ;valor do TROCO a dar é passado para R4

        RET
;------FIM ROTINA CALCULAR TROCO------

;---------------------------;
;   ROTINA DEVOLVER TROCO   ;
;---------------------------;
```

```
;devolve o troco


DevolverTroco:

;R4 TROCO a dar


        MOV A, R4       ;move valor do TROCO a dar para o acumulador (JZ verifica o valor de A)

        JZ FIM_DEVOLVERTROCO        ;se TROCO a dar é igual a zero, salta para o fim da rotina


checkTroco50:

        MOV A, StockM50             ;move valor do Stock de moedas de 0,50 euros para o
acumulador (JZ verifica o valor de A)

        JZ checkTroco10             ;se Stock for igual a zero, salta para o ciclo de verificar as
moedas de 0,10 euros

        CJNE R4, #50, notEQ50   ;esta operacao é executada com o intuito de verificar o valor da flag de
carry

        notEQ50:

        JC checkTroco10             ;se C = 1, TROCO a dar < 0,50 euros, salta para o ciclo de
verificar as moedas de 0,10 euros

                                                    ;se C = 0, TROCO a dar >= 0,50 euros , é
possivel devolver moeda de 0,50, sao executadas as proximas instruções


        CALL mydelay

        CALL RodarMotor50       ;roda o motor das moedas de 0,50 euros, devolvendo uma moeda

        CALL mydelay


        MOV A, R4                   ;move TROCO a dar(R4) para o acumulador

        CLR C                       ;limpa o valor da flag de carry (pois é usado na operação
SUBB)

        SUBB A, #50                 ;decrementa valor da moeda de 0,50 euros ao acumulador

        MOV R4, A                   ;actualiza o valor do TROCO a dar (R4)


        DEC StockM50            ;decrementa stock das moedas de 0,50 euros

        JMP checkTroco50        ;salta para o inicio do ciclo
```

```
checkTroco10:

        MOV A, StockM10                 ;move valor do Stock de moedas de 0,10 euros para
o acumulador (JZ verifica o valor de A)

        JZ checkTroco05                 ;se Stock for igual a zero, salta para o ciclo de verificar as
moedas de 0,05 euros

        CJNE R4, #10, notEQ10   ;esta operacao é executada com o intuito de verificar o valor da flag de
carry

        notEQ10:

        JC checkTroco05                 ;se C = 1, TROCO a dar < 0,10 euros, salta para o ciclo de
verificar as moedas de 0,05 euros

                                                ;se C = 0, TROCO a dar >= 0,10 euros, é
possivel devolver moeda de 0,10, sao executadas as proximas instruções


        CALL mydelay

        CALL RodarMotor10       ;roda o motor das moedas de 0,10 euros, devolvendo uma moeda

        CALL mydelay


        MOV A, R4                       ;move TROCO a dar(R4) para o acumulador

        CLR C                           ;limpa o valor da flag de carry (pois é usado na operação
SUBB)

        SUBB A, #10                     ;decrementa valor da moeda de 0,10 euros ao acumulador

        MOV R4, A                       ;actualiza o valor do TROCO a dar (R4)


        DEC StockM10            ;decrementa stock das moedas de 0,10 euros

        JMP checkTroco10        ;salta para o inicio do ciclo


checkTroco05:

        MOV A, StockM05                 ;move valor do Stock de moedas de 0,05 euros para
o acumulador (JZ verifica o valor de A)

        JZ FIM_DEVOLVERTROCO    ;se Stock for igual a zero, salta para o fim da rotina

        CJNE R4, #5, notEQ05    ;esta operacao é executada com o intuito de verificar o valor da flag de
carry

        notEQ05:
```

```
        JC FIM_DEVOLVERTROCO      ;se C = 1, TROCO a dar < 0,05 euros, salta para o fim da
rotina
                                                    ;se C = 0, TROCO a dar >= 0,05 euros, é
possivel devolver moeda de 0,05, sao executadas as proximas instruções

        CALL mydelay
        CALL RodarMotor05     ;roda o motor das moedas de 0,05 euros, devolvendo uma moeda
        CALL mydelay

        MOV A, R4             ;move TROCO a dar(R4) para o acumulador
        CLR C                 ;limpa o valor da flag de carry (pois é usado na operação
SUBB)
        SUBB A, #5            ;decrementa valor da moeda de 0,05 euros ao acumulador
        MOV R4, A             ;actualiza o valor do TROCO a dar (R4)

        DEC StockM05          ;decrementa stock das moedas de 0,05 euros
        JMP checkTroco05      ;salta para o inicio do ciclo

FIM_DEVOLVERTROCO:

;se não for possivel continuar a dar troco devido à falta de stock de moedas a rotina termina (mesmo que
o valor do troco a dar não chegue a 0)
;desta forma o utilizador não vai receber todo o troco que tem direito mas vai receber a bebida escolhida

        RET
;------FIM ROTINA DEVOLVER TROCO------

;-----------------------;
;   ROTINA DAR BEBIDA   ;
;-----------------------;
;dá a bebida escolhida
```

```
DarBebida:

;R3 contem a BEBIDA escolhida


CASOB1:

        CJNE R3, #1, CASOB2    ;verifica se a BEBIDA escolhida foi a bebida1, se não salta para o
CASOB2


        CALL RodarMotorB1      ;roda o motor da bebida1, devolvendo a bebida1


        DEC StockB1                    ;decrementa o stock da bebida1

        JMP FIM_DAR_BEBIDA             ;salta para o fim da rotina


CASOB2:

        CJNE R3, #2, CASOB3    ;verifica se a BEBIDA escolhida foi a bebida2, se não salta para o
CASOB3


        CALL RodarMotorB2      ;roda o motor da bebida2, devolvendo a bebida2


        DEC StockB2                    ;decrementa o stock da bebida2

        JMP FIM_DAR_BEBIDA             ;salta para o fim da rotina


CASOB3:

        CJNE R3, #3, CASOB4 ;verifica se a BEBIDA escolhida foi a bebida3, se não salta para o
CASOB4


        CALL RodarMotorB3      ;roda o motor da bebida3, devolvendo a bebida3


        DEC StockB3                    ;decrementa o stock da bebida3

        JMP FIM_DAR_BEBIDA             ;salta para o fim da rotina
```

```
CASOB4:
        CJNE R3, #4, FIM_DAR_BEBIDA ;verifica se a BEBIDA escolhida foi a bebida4, se não salta para o fim da rotina

        CALL RodarMotorB4       ;roda o motor da bebida4, devolvendo a bebida4

        DEC StockB4             ;decrementa o stock da bebida4
        JMP FIM_DAR_BEBIDA      ;salta para o fim da rotina

FIM_DAR_BEBIDA:

        RET

;------FIM ROTINA DAR BEBIDA------

;------------------------------;
;   ROTINAS DE INICIALIZACAO   ;
;------------------------------;

;inicializa stocks de bebidas e moedas
InicializarStocks:

        MOV StockM05, #150      ;valores dos stocks das moedas
        MOV StockM10, #150
        MOV StockM20, #150
        MOV StockM50, #150

        MOV StockB1, #25        ;valores dos stocks das bebidas
        MOV StockB2, #25
```

```
        MOV StockB3, #25
        MOV StockB4, #25

        RET

;inicializa valores das portas
InicializarPortas:

        MOV Pselecao, #0xFF     ;inicializaçao dos botoes das moedas e bebidas - lógica negada

        SETB Cancel             ;inicializacao do botao Cancel - lógica negada

        SETB SensorM05          ;inicialização dos sensores dos motores das moedas - lógica negada
        SETB SensorM10
        SETB SensorM50

        CLR MotorM05  ;inicialização dos motores das moedas - lógica normal
        CLR MotorM10
        CLR MotorM50

        SETB SensorB1  ;inicialização dos sensores dos motores das bebidas - lógica negada
        SETB SensorB2
        SETB SensorB3
        SETB SensorB4

        CLR MotorB1             ;inicialização dos motores das bebidas - lógica normal
        CLR MotorB2
        CLR MotorB3
        CLR MotorB4
```

```
        RET


;activar transmissao da porta serial

ActivarTransPortaSerial:


        MOV SCON, #0x50         ;Porta serial a funcionar em modo 1 (8-bit UART) (0101 0000b ou 80d) SFR SCON - SM0 = 0, SM1 = 1 , REN = 1

        MOV TMOD, #0x20         ;utilizar o Timer 1 no modo 2 (0001 0000b ou 32d) SFR TMOD - C/T = 0(usar como timer), M1 = 1, M0 = 0 (do lado so timer1, 4 bits mais significativos do SFR TMOD)

        MOV TH1, #0xFD          ;9600 Bps  at 11.059MHz (taxa de transmissão)

        MOV TL1, #0xFD          ;timer de 8 bits

        SETB TR1                ;"liga" o timer1 - TR1 está no SFR TCON

        SETB TI                         ;inicialização da flag de emissão


        RET


;inicializar interrupções

Interrupcoes_init:


        MOV IP, #00000001b ;interrupção externa 0 tem a prioridade mais elevada, SFR IP - PX0 = 1

        MOV IE, #10000001b ;ativa interrupçao externa 0, SFR IE - EA = 1 (habilita o uso de interrupções) , EX0 = 1 (habilita o uso da interrupção externa0)

        SETB IT0        ;a interrupção externa 0 vai ser detetada na transição descendente (1 para 0) IT0 é um bit do SFR TCON


        RET


;--------FIM ROTINAS DE INICIALIZACAO-----------




;---------------------;

;   ROTINA TXD_DATA   ;
```

```
;---------------------;

;escreve no display o valor total introduzido


TXD_data:

;R1 contem o valor TOTAL de moedas introduzidas

cicloData:

        JNB TI, cicloData ;Enquanto T1 não for 1 não executa as proximas instruçoes (emissão prévia ainda a decorrer)

        CLR TI                  ;Limpa a flag de emissão (TI) para a proxima emissão (tem de ser feito por software)

        MOV SBUF, R1    ;coloca o valor(R1) no buffer (SFR SBUF)


        RET

;------FIM ROTINA TXD_DATA------




;-------------------------------------;

;   ROTINA DA INTERRUPCAO EXTERNA 0   ;

;-------------------------------------;

;tratamento da Interrupçao externa 0 (botão cancel)


InterrupcaoExt0:

;R0 contem o valor do ESTADO


        CJNE R0, #1, estadoNot1 ;Verificamos se o estado é igual a 1, se não for a instrução a baixo não é executada

                                                        ;pois só faz sentido cancelar quando estamos no estado de introduçao de moedas/seleçao bebida (estado 1)


        MOV R0, #3                      ;Muda para ESTADO 3


        estadoNot1:
```

```
        MOV Pselecao, #0xFF             ;Reset dos pinos de seleção de moedas e bebidas
        SETB Cancel                     ;Reset do pino do botão cancel

        RETI

;------FIM DA ROTINA DA INTERRUPCAO----------


;------------------------------;
;  ROTINAS PARA RODAR MOTORES   ;
;------------------------------;

RodarMotor50:

        SensorM50_1:
        JNB SensorM50, SensorM50_0      ;se o sensor for activado (0), salta para o proximo ciclo
        SETB MotorM50                   ;roda o motor das moedas de 0,50 euros
        JMP SensorM50_1                 ;salta para o inicio do ciclo

        ;roda o motor um pouco para este deixar de activar o sensor
        SensorM50_0:
        JB SensorM50, FimMotorM50       ;se o sensor for desactivado (1), sai do ciclo
        SETB MotorM50                   ;roda o motor das moedas de 0,50 euros
        JMP SensorM50_0                 ;salta para o inicio do ciclo

        FimMotorM50:
        CLR MotorM50                    ;para o motor das moedas de 0,50 euros

        RET
;-----------------------------------
```

```
RodarMotor10:

        SensorM10_1:
        JNB SensorM10, SensorM10_0     ;se o sensor for activado (0), salta para o proximo ciclo
        SETB MotorM10                                   ;roda o motor das moedas de 0,10 euros
        JMP SensorM10_1                                 ;salta para o inicio do ciclo

        ;roda o motor um pouco para este deixar de activar o sensor
        SensorM10_0:
        JB SensorM10, FimMotorM10      ;se o sensor for desactivado (1), sai do ciclo
        SETB MotorM10                                   ;roda o motor das moedas de 0,10 euros
        JMP SensorM10_0                                 ;salta para o inicio do ciclo

        FimMotorM10:
        CLR MotorM10                            ;para o motor das moedas de 0,10 euros

        RET
;----------------------------------

RodarMotor05:

        SensorM05_1:
        JNB SensorM05, SensorM05_0     ;se o sensor for activado (0), salta para o proximo ciclo
        SETB MotorM05                                   ;roda o motor das moedas de 0,05 euros
        JMP SensorM05_1                                 ;salta para o inicio do ciclo

        ;roda o motor um pouco para este deixar de activar o sensor
        SensorM05_0:
        JB SensorM05, FimMotorM05      ;se o sensor for desactivado (1), sai do ciclo
```

```
        SETB MotorM05                   ;roda o motor das moedas de 0,05 euros
        JMP SensorM05_0                 ;salta para o inicio do ciclo

        FimMotorM05:
        CLR MotorM05                ;para o motor das moedas de 0,05 euros

        RET
;-----------------------------------

RodarMotorB1:

        SensorB1_1:
        JNB SensorB1, SensorB1_0        ;se o sensor for activado (0), salta para o proximo ciclo
        SETB MotorB1                    ;roda o motor da bebida1
        JMP SensorB1_1                  ;salta para o inicio do ciclo

        ;roda o motor um pouco para este deixar de activar o sensor
        SensorB1_0:
        JB SensorB1, FimMotorB1         ;se o sensor for desactivado (1), sai do ciclo
        SETB MotorB1                    ;roda o motor da bebida1
        JMP SensorB1_0                  ;salta para o inicio do ciclo

        FimMotorB1:
        CLR MotorB1                     ;para o motor da bebida1

        RET
;-----------------------------------

RodarMotorB2:
```

```
        SensorB2_1:
        JNB SensorB2, SensorB2_0        ;se o sensor for activado (0), salta para o proximo ciclo
        SETB MotorB2                    ;roda o motor da bebida2
        JMP SensorB2_1                  ;salta para o inicio do ciclo

        ;roda o motor um pouco para este deixar de activar o sensor
        SensorB2_0:
        JB SensorB2, FimMotorB2         ;se o sensor for desactivado (1), sai do ciclo
        SETB MotorB2                    ;roda o motor da bebida2
        JMP SensorB2_0                  ;salta para o inicio do ciclo

        FimMotorB2:
        CLR MotorB2                             ;para o motor da bebida2

        RET
;-----------------------------------

RodarMotorB3:

        SensorB3_1:
        JNB SensorB3, SensorB3_0        ;se o sensor for activado (0), salta para o proximo ciclo
        SETB MotorB3                    ;roda o motor da bebida3
        JMP SensorB3_1                  ;salta para o inicio do ciclo

        ;roda o motor um pouco para este deixar de activar o sensor
        SensorB3_0:
        JB SensorB3, FimMotorB3         ;se o sensor for desactivado (1), sai do ciclo
        SETB MotorB3                    ;roda o motor da bebida3
        JMP SensorB3_0                  ;salta para o inicio do ciclo
```

```
        FimMotorB3:
        CLR MotorB3                                 ;para o motor da bebida3

        RET
;----------------------------------

RodarMotorB4:

        SensorB4_1:
        JNB SensorB4, SensorB4_0        ;se o sensor for activado (0), salta para o proximo ciclo
        SETB MotorB4                    ;roda o motor da bebida4
        JMP SensorB4_1                  ;salta para o inicio do ciclo

        ;roda o motor um pouco para este deixar de activar o sensor
        SensorB4_0:
        JB SensorB4, FimMotorB4         ;se o sensor for desactivado (1), sai do ciclo
        SETB MotorB4                    ;roda o motor da bebida4
        JMP SensorB4_0                  ;salta para o inicio do ciclo

        FimMotorB4:
        CLR MotorB4                                 ;para o motor da bebida4

        RET

;--------FIM ROTINAS PARA RODAR MOTORES----------


;-----------------------------;
;   ROTINA CALIBRAR MOTORES   ;
;-----------------------------;
```

```
;chama as funçoes que rodam os motores, deixando-os calibrados na posição imediatamente à "frente" do respectivo sensor


CalibrarMotores:


        CALL RodarMotor50

        CALL RodarMotor10

        CALL RodarMotor05

        CALL RodarMotorB1

        CALL RodarMotorB2

        CALL RodarMotorB3

        CALL RodarMotorB4


        RET


;--------FIM ROTINA CALIBRAR MOTORES----------

END
```

6. Anexo C – Código em linguagem C

```
#include <reg51.h>


//Prototipos das funçoes


void inicializarStocks(void);

void inicalizarPortas(void);

void ler_valor(unsigned int);

void TXD_data(unsigned char);

void ler_bebida(void);

void mydelay(unsigned int);

void checkTotalCusto(unsigned int, unsigned int);

void devolverTroco(unsigned int);

void darBebida(unsigned int);

void activarTransPortaSerial();

unsigned int calcularTroco(unsigned int, unsigned int);

void interrupcoes_init(void);

void external_int0(void);

void rodarMotor50(void);

void rodarMotor10(void);

void rodarMotor05(void);

void rodarMotorB1(void);

void rodarMotorB2(void);

void rodarMotorB3(void);

void rodarMotorB4(void);

void calibrarMotores(void);


//declaração das variaveis globais


unsigned int estado;                    //determina em que estado está o programa
```

```
unsigned int total;                    //total de dinheiro introduzido em centimos
unsigned int custo;                    //custo da bebida selecionada
unsigned int bebida;          //bebida selecionada

unsigned int stock05;         //stock de moedas de 0,05
unsigned int stock10;         //stock de moedas de 0,10
unsigned int stock20;         //stock de moedas de 0,20
unsigned int stock50;         //stock de moedas de 0,50

unsigned int stockb1;         //Stock da bebida1
unsigned int stockb2;         //Stock da bebida2
unsigned int stockb3;         //Stock da bebida3
unsigned int stockb4;         //Stock da bebida4

//definiçao dos Pinos

sbit sensor10 = P0^0;         //Pino do sensor do motor da moeda de 0,10 euros
sbit sensor50 = P0^1;         //Pino do sensor do motor da moeda de 0,50 euros
sbit sensor05 = P0^2;         //Pino do sensor do motor da moeda de 0,05 euros

sbit motor10 = P2^4;          //Pino do motor da moeda de 0,10 euros
sbit motor50 = P2^5;          //Pino do motor da moeda de 0,50 euros
sbit motor05 = P2^6;          //Pino do motor da moeda de 0,05 euros

sbit sensorb1 = P2^0;         //Pino do sensor do motor da bebida 1
sbit sensorb2 = P2^1;         //Pino do sensor do motor da bebida 2
sbit sensorb3 = P2^2;         //Pino do sensor do motor da bebida 3
sbit sensorb4 = P2^3;         //Pino do sensor do motor da bebida 4
```

```
sbit motorb1 = P3^4;            //Pino do motor da bebida 1
sbit motorb2 = P3^5;            //Pino do motor da bebida 2
sbit motorb3 = P3^6;            //Pino do motor da bebida 3
sbit motorb4 = P3^7;            //Pino do motor da bebida 4

sbit cancel = P3^2;                     //Pino do botão Cancel

sbit pserial = P3^1;            //Pino da porta serial

//definição das constantes

#define custob1 80              //Custo da bebida1
#define custob2 100             //Custo da bebida2
#define custob3 120             //Custo da bebida3
#define custob4 150             //Custo da bebida4

void main (void)
{
        inicializarStocks();                            //inicializa stocks de bebidas e moedas
        inicalizarPortas();                                     //inicializa valores das portas
        activarTransPortaSerial(); //activar transmissao da porta serial
        interrupcoes_init();                            //inicializar interrupções
        calibrarMotores();                                      //calibra os motores das bebidas e
das moedas

        estado = 1;             //programa começa no estado 1

        total = 0;              //valor total introduzido começa em 0
        custo = 0;              //custo da bebida selecionada é 0 no inicio
        bebida = 0;             //no inicio não há nenhuma bebida escolhida
```

```
for(;;)		//ciclo infinito
{

	switch(estado)
	{
		//estado de introdução de moedas e seleção de bebida
		case 1:
			ler_valor(total);
	//lê e acumula valor das moedas introduzidas
			TXD_data(total);
	//escreve no display o valor total introduzido
			ler_bebida();
		//lê a bebida escolhida e guarda o custo da mesma
			checkTotalCusto(total, custo);	//Verifica se o valor total de moedas introduzidas é maior que o custo da bebida escolhida, se for muda para o estado 2
			break;

		//estado de venda, devolve troco e dá a bebida escolhida
		case 2:
			devolverTroco(calcularTroco(total, custo)); //calcula o troco a dar e devolve esse mesmo troco
			darBebida(bebida);
								//dá a bebida escolhida
			custo = 0;
		//reset da variavel custo da bebida
			total = 0;
		//reset do total de dinheiro introduzido
			bebida = 0;
		//reset da variavel da bebida selecionada
			estado = 1;
		//muda para o estado 1 para voltar a executar as funçoes desse mesmo estado
			P1=0xFF;
		//Reset dos pinos de seleção de moedas e bebidas
			break;
```

```
                        //estado executado após o botão cancel ser pressionado

                        case 3:

                                devolverTroco(total);           //devolve o dinheiro total
introduzido

                                custo = 0;
        //reset da variavel custo da bebida

                                total = 0;
        //reset do total de dinheiro introduzido

                                bebida = 0;
        //reset da variavel da bebida selecionada

                                estado = 1;
        //muda para o estado 1 para voltar a executar as funçoes desse mesmo estado

                                P1=0xFF;
        //Reset dos pinos de seleção de moedas e bebidas

                                break;


                }


        }


}


void ler_valor(unsigned int valor)

{


        int y = 0;

        y = (P1&0x0F);   //introduçao de uma mascara, executa uma operaçao AND bit a bit, passando
para 0 os valores dos pinos referentes às bebidas

                                                                        //desta forma sao apenas
lidos os pinos referentes às moedas


        switch(y)                               //Y contem valores dos pinos referentes às moedas
(os das bebidas estao a 0), salta para um caso mediante a combinação de bits
```

```
        {                                                   //cada combinação de bits
equivale à seleção de uma das moedas, esta combinação está em decimal

                case 15:                                                    //0000 1111
                        valor = valor + 0;          //Não foram introduzidas moedas, somamos 0
                        P1=0xFF;                                            //Reset
dos pinos de seleção de moedas e bebidas
                        break;

                case 14:                                                    //0000 1110
                        valor = valor + 5;          //Introduzida moeda 5 centimos, somamos 5
                        stock05++;                                          //Adiciona uma
moeda ao stock de 5 centimos
                        P1=0xFF;                                            //Reset
dos pinos de seleção de moedas e bebidas
                        mydelay(20000);
                        break;

                case 13:                                                    //0000 1101
                        valor = valor + 10;             //Introduzida moeda 10 centimos, somamos
10
                        stock10++;                                          //Adiciona uma
moeda ao stock de 10 centimos
                        P1=0xFF;                                            //Reset
dos pinos de seleção de moedas e bebidas
                        mydelay(20000);
                        break;

                case 11:                                                    //0000 1011
                        valor = valor + 20;             //Introduzida moeda 20 centimos, somamos
20
                        stock20++;                                          //Adiciona uma
moeda ao stock de 20 centimos
```

```
                P1=0xFF;                                                    //Reset
dos pinos de seleção de moedas e bebidas

                mydelay(20000);

                break;


            case 7:                                                         //0000
0111

                valor = valor + 50;             //Introduzida moeda 50 centimos, somamos
50

                stock50++;                                                  //Adiciona uma
moeda ao stock de 50 centimos

                P1=0xFF;                                                    //Reset
dos pinos de seleção de moedas e bebidas

                mydelay(20000);

                break;

        }


        total = valor;


}


void ler_bebida(void)

{


        int y = 0;

        y = (P1&0xF0);   //introduçao de uma mascara, executa uma operaçao AND bit a bit, passando
para 0 os valores dos pinos referentes às moedas

                                                                            //desta forma sao apenas
lidos os pinos referentes às bebidas

        bebida = 0;

        custo = 0;


        switch(y)                           //Y contem valores dos pinos referentes às bebidas
(os das moedas estao a 0), salta para um caso mediante a combinação de bits
```

```
        {                                                   //cada combinação de bits
equivale à seleção de uma das bebidas, esta combinação está em decimal


                case 240:                                   //1111
0000
                        bebida = 0;                         //Não foi
selecionada nenhuma bebida
                        custo = 0;                          //Custo é zero

                        P1=0xFF;                            //Reset
dos pinos de seleção de moedas e bebidas
                        break;


                case 224:                                   //1110
0000
                        if(stockb1 > 0)                     //caso exista a bebida1 em stock
                        {
                                bebida = 1;                 //Selecionada
bebida1
                                custo = custob1;            //Custo da bebida escolhida passa a ter o
valor do custo da bebida1
                        }
                        P1=0xFF;                            //Reset
dos pinos de seleção de moedas e bebidas
                        mydelay(20000);
                        break;


                case 208:                                   //1101
0000
                        if(stockb2 > 0)                     //caso exista a bebida2 em stock
                        {
                                bebida = 2;                 //Selecionada
bebida2
                                custo = custob2;            //Custo da bebida escolhida passa a ter o
valor do custo da bebida2
```

```
                }
                P1=0xFF;                                        //Reset
dos pinos de seleção de moedas e bebidas
                mydelay(20000);
                break;

        case 176:                                               //1011
0000
                if(stockb3 > 0)                 //caso exista a bebida3 em stock
                {
                        bebida = 3;                             //Selecionada
bebida3
                        custo = custob3;        //Custo da bebida escolhida passa a ter o
valor do custo da bebida3
                }
                P1=0xFF;                                        //Reset
dos pinos de seleção de moedas e bebidas
                mydelay(20000);
                break;

        case 112:                                               //0111
0000
                if(stockb4 > 0)                 //caso exista a bebida4 em stock
                {
                        bebida = 4;                             //Selecionada
bebida4
                        custo = custob4;        //Custo da bebida escolhida passa a ter o
valor do custo da bebida4
                }
                P1=0xFF;                                        //Reset
dos pinos de seleção de moedas e bebidas
                mydelay(20000);
                break;
    }
```

```
}

void checkTotalCusto(unsigned int total, unsigned int custo)

{

        if(total >= custo && bebida != 0 && estado != 3) //verifica se o dinheiro inserido é maior ou igual ao custo da bebida selecionada e se existe alguma bebida selecionada

        {

         //Estado nao pode ser igual a 3 porque significa que o botão cancel foi pressionado e as proximas funçoes a executar sao as do estado 3


                estado = 2;                                     //muda para o estado 2 para serem executadas as funçoes desse mesmo estado (estado de venda)

        }

}


void mydelay(unsigned int delay) //função que gera um atraso

{

        unsigned int i;

        for (i=0; i<delay; i++);                        //a função acaba quando "contar" de 0 até ao valor do delay

}


unsigned int calcularTroco(unsigned int total, unsigned int custo)

{

        int troco = 0;

        troco = total - custo; //troco a dar é o valor do total de dinheiro inserido menos o valor do custo da bebida selecionada


        return troco;                                   //retorna valor do troco a dar

}
```

```
void devolverTroco(unsigned int troco)

{

		if(troco == 0)	//caso o troco seja 0 (não é preciso dar troco)

		{

			return;					//sai da função

		}

		else

		{

			while(stock50 > 0 && troco >= 50)			//verifica se existe stock de moedas de 0,50 e se troco a dar é igual ou maior a 50 cents

			{

				mydelay(20000);

				rodarMotor50();
			//roda o motor das moedas de 0,50 euros, devolvendo uma moeda

				mydelay(20000);



				stock50--;
					//decrementa stock das moedas de 0,50 euros

				troco = troco - 50;
		//decrementa o valor da moeda de 0,50 euros ao troco a dar

				TXD_data(troco);

			}



			while(stock10 > 0 && troco >= 10)			//verifica se existe stock de moedas de 0,10 e se troco a dar é igual ou maior a 10 cents

			{

				mydelay(20000);

				rodarMotor10();
			//roda o motor das moedas de 0,10 euros, devolvendo uma moeda

				mydelay(20000);



				stock10--;
					//decrementa stock das moedas de 0,10 euros
```

```
                troco = troco - 10;
            //decrementa o valor da moeda de 0,10 euros ao troco a dar

                TXD_data(troco);

            }


            while(stock05 > 0 && troco >= 5)        //verifica se existe stock de moedas
de 0,05 e se troco a dar é igual ou maior a 05 cents

            {

                mydelay(20000);

                rodarMotor05();
            //roda o motor das moedas de 0,05 euros, devolvendo uma moeda

                mydelay(20000);



                stock05--;
                    //decrementa stock das moedas de 0,05 euros

                troco = troco - 5;
        //decrementa o valor da moeda de 0,05 euros ao troco a dar

                TXD_data(troco);

            }

        }


        //se não for possivel continuar a dar troco devido à falta de stock de moedas a função
termina (mesmo que o valor do troco a dar não chegue a 0)

        //desta forma o utilizador não vai receber todo o troco que tem direito mas vai receber a
bebida escolhida

}


void darBebida(unsigned int bebida)

{

    switch(bebida)          //mediante o valor da variavel bebida, executa um dos casos. Cada
caso corresponde a uma das quatro bebidas

    {

        case 1:                                         //foi escolhida a bebida1
```

```
                rodarMotorB1();         //roda o motor da bebida1, devolvendo a bebida1
                stockb1--;                      //decrementa o stock da bebida1
                break;

            case 2:                                 //foi escolhida a bebida2

                rodarMotorB2();         //roda o motor da bebida2, devolvendo a bebida2
                stockb2--;                      //decrementa o stock da bebida2
                break;

            case 3:                                 //foi escolhida a bebida3

                rodarMotorB3();         //roda o motor da bebida3, devolvendo a bebida3
                stockb3--;                      //decrementa o stock da bebida3
                break;

            case 4:                                 //foi escolhida a bebida4

                rodarMotorB4();         //roda o motor da bebida4, devolvendo a bebida4
                stockb4--;                      //decrementa o stock da bebida4
                break;
        }

}

void inicializarStocks(void)
{
        stock05 = 150;          //valores dos stocks das moedas
        stock10 = 150;
        stock20 = 150;
```

```
	stock50 = 150;

	stockb1 = 25;		//valores dos stocks das bebidas
	stockb2 = 25;
	stockb3 = 25;
	stockb4 = 25;
}

void inicalizarPortas(void)
{

	P1 = 0xFF;			//incialização dos botoes das moedas e e bebidas - lógica negada

	cancel = 1;			//inicialização do pino do botão Cancel - lógica negada

	sensor10 = 1;		//inicialização dos sensores dos motores das moedas - lógica negada
	sensor50 = 1;
	sensor05 = 1;

	motor10 = 0;		//inicialização dos motores das moedas - lógica normal
	motor50 = 0;
	motor05 = 0;

	sensorb1 = 1;		//inicialização dos sensores dos motores das bebidas - lógica negada
	sensorb2 = 1;
	sensorb3 = 1;
	sensorb4 = 1;

	motorb1 = 0;		//inicialização dos motores das bebidas - lógica normal
	motorb2 = 0;
```

```
	motorb3 = 0;

	motorb4 = 0;


}


void activarTransPortaSerial(void)

{

	SCON = 0x50;	//Porta serial a funcionar em modo 1 (8-bit UART) (0101 0000b ou 80d) SFR SCON - SM0 = 0, SM1 = 1 , REN = 1

	TMOD = 0x20;	//utilizar o Timer 1 no modo 2 (0001 0000b ou 32d) SFR TMOD - C/T = 0(usar como timer), M1 = 1, M0 = 0 (do lado so timer1, 4 bits mais significativos do SFR TMOD)

	TH1 = 0xFD;	//9600 Bps  at 11.059MHz (taxa de transmissão)

	TL1 = 0xFD;	//timer de 8 bits

	TR1 = 1;	//"liga" o timer1 - TR1 está no SFR TCON

	TI = 1;	//inicialização da flag de emissão

}


void interrupcoes_init(void)

{

	IP	= 1;		//interrupção externa 0 tem a prioridade mais elevada (0000 0001b ou 0x01), SFR IP - PX0 = 1

	IE	= 129;	//ativa interrupçao externa 0 (1000 0001b ou 0x81), SFR IE - EA = 1 (habilita o uso de interrupções) , EX0 = 1 (habilita o uso da interrupção externa0)

	IT0 = 1;	//a interrupção externa 0 vai ser detetada na transição descendente (1 para 0) IT0 é um bit do SFR TCON

}


void TXD_data(unsigned char value)

{


	while (TI!=1){};	//Enquanto T1 não for 1 não executa as proximas instruçoes (emissão prévia ainda a decorrer)
```

```
	TI=0;		//Limpa a flag de emissão (TI) para a proxima emissão (tem de ser feito por software)
	SBUF = value;	//coloca value no buffer (SFR SBUF)

}

void external_int0(void) interrupt 0		//função de tratamento da interrupção externa 0, função é executada quando o botão cancelar é pressionado
{

	if(estado == 1) //só faz sentido	cancelar quando estamos no estado de introduçao de moedas/seleçao bebida
	{
		estado = 3;	//Muda para estado 3
	}
	P1 = 0xFF;			//Reset dos pinos de seleção de moedas e bebidas
	cancel = 1;			//Reset do pino do botão cancel

}

void rodarMotor50(void)
{
				while(sensor50 == 1)	//enquanto o sensor do motor das moedas de 0,50 euros não for activado
				{
					motor50 = 1;				//roda o motor das moedas de 0,50 euros
				}

				//roda o motor um pouco para este deixar de activar o sensor
				while(sensor50 == 0)	//enquanto o sensor do motor das moedas de 0,50 euros está activado
				{
```

```
                motor50 = 1;                    //roda o motor
das moedas de 0,50 euros
            }
            motor50 = 0;                        //para o motor
das moedas de 0,50 euros
}

void rodarMotor10(void)
{
            while(sensor10 == 1)    //enquanto o sensor do motor das moedas
de 0,10 euros não for activado
            {
                motor10 = 1;                    //roda o motor
das moedas de 0,10 euros
            }

            //roda o motor um pouco para este deixar de activar o sensor
            while(sensor10 == 0)    //enquanto o sensor do motor das moedas
de 0,10 euros está activado
            {
                motor10 = 1;                    //roda o motor
das moedas de 0,10 euros
            }
            motor10 = 0;                        //para o motor
das moedas de 0,10 euros
}

void rodarMotor05(void)
{
            while(sensor05 == 1)    //enquanto o sensor do motor das moedas
de 0,05 euros não for activado
            {
                motor05 = 1;                    //roda o motor
das moedas de 0,05 euros
```

```
                    }

                    //roda o motor um pouco para este deixar de activar o sensor
                    while(sensor05 == 0)        //enquanto o sensor do motor das moedas
de 0,05 euros está activado
                    {
                        motor05 = 1;                                    //roda o motor
das moedas de 0,05 euros
                    }
                    motor05 = 0;                                        //para o motor
das moedas de 0,05 euros
}

void rodarMotorB1(void)
{
                while(sensorb1 == 1)        //enquanto o sensor do motor da bebida1 não for
activado
                {
                    motorb1 = 1;                                //roda o motor da bebida1
                }

                //roda o motor um pouco para este deixar de activar o sensor
                while(sensorb1 == 0)        //enquanto o sensor do motor da bebida1 está
activado
                {
                    motorb1 = 1;                                //roda o motor da bebida1
                }
                motorb1 = 0;                                    //para o motor da bebida1
}


void rodarMotorB2(void)
```

```
{
                while(sensorb2 == 1)       //enquanto o sensor do motor da bebida2 não for
activado
                {
                        motorb2 = 1;                                    //roda o motor da bebida2
                }

                //roda o motor um pouco para este deixar de activar o sensor
                while(sensorb2 == 0)       //enquanto o sensor do motor da bebida2 está
activado
                {
                        motorb2 = 1;                                    //roda o motor da bebida2
                }
                motorb2 = 0;                                            //para o motor da bebida2
}

void rodarMotorB3(void)
{
                while(sensorb3 == 1)       //enquanto o sensor do motor da bebida3 não for
activado
                {
                        motorb3 = 1;                                    //roda o motor da bebida3
                }

                //roda o motor um pouco para este deixar de activar o sensor
                while(sensorb3 == 0)       //enquanto o sensor do motor da bebida3 está
activado
                {
                        motorb3 = 1;                                    //roda o motor da bebida3
                }
                motorb3 = 0;                                            //para o motor da bebida3
}
```

```
void rodarMotorB4(void)
{
                    while(sensorb4 == 1)       //enquanto o sensor do motor da bebida4 não for
activado
                    {
                              motorb4 = 1;                              //roda o motor da bebida4
                    }

                    //roda o motor um pouco para este deixar de activar o sensor
                    while(sensorb4 == 0)       //enquanto o sensor do motor da bebida4 está
activado
                    {
                              motorb4 = 1;                              //roda o motor da bebida4
                    }
                    motorb4 = 0;                                        //para o motor da bebida4
}

void calibrarMotores(void)
{
          //chama as funçoes que rodam os motores, deixando-os calibrados na posição imediatamente à
"frente" do respectivo sensor

          rodarMotor50();
          rodarMotor10();
          rodarMotor05();
          rodarMotorB1();
          rodarMotorB2();
          rodarMotorB3();
          rodarMotorB4();
}
```

```
//void rodarMotor(sbit motor, sbit sensor)
//{
//        while(sensor == 1) //enquanto o sensor for 1, roda o motor
//        {
//                motor = 1;
//        }
//
//        while(sensor == 0) //roda o motor um pouco, para este deixar de activar o sensor
//        {
//                motor = 1;
//        }
//
//        motor = 0;                  //para o motor
//}
```