# Trabalho de Eletrônica Embarcada

1

Gerado por Doxygen 1.9.1

# Chapter 1

# Índice dos ficheiros

## 1.1 Lista de ficheiros

Lista de todos os ficheiros com uma breve descrição:

# Chapter 2

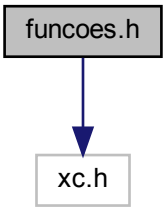
# Documentação do ficheiro

## 2.1 Referência ao ficheiro funcoes.h

Declaração das funções de controle, aquisição de dados e movimentação do motor.

```
#include <xc.h>
```

Diagrama de dependências de inclusão para funcoes.h:




### Funções

- uint16_t velocidade_som ()

  *Retorna a velocidade do som de acordo com a temperatura.*
- void funcao_aumento_num_passos ()

  *Realiza a mudança dos passos no sentido original.*
- void funcao_diminui_num_passos ()

  *Realiza a mudança dos passos no sentido contrário ao original.*
- void Passo1 ()

  *Aciona o primeiro passo do motor.*
- void Passo2 ()

  *Aciona o segundo passo do motor.*
- void Passo3 ()

  *Aciona o terceiro passo do motor.*

- void Passo4 ()

  *Aciona o quarto passo do motor.*
- void DRMotordePassos ()

  *Atualiza os passos do motor de acordo com a posição desejada.*
- void posicao_bola_tubo (void)

  *Calcula a posição da bola no tubo.*
- void media (uint16_t timer)

  *Calcula a média do sistema com base no tempo medido.*
- void medir_altura (void)

  *Realiza a medição da altura.*
- void Posicao_Bola_Tubo (void)

  *Adquire a posição da bola no tubo.*
- void EnviaTx (void)

  *Carrega o buffer de transmissão (Tx) com as informações desejadas.*
- void selecao_do_modo (void)

  *Seleciona o modo de operação do sistema e configura os parâmetros correspondentes.*
- void modo_ventoinha (void)

  *Controla a ventoinha para ajustar a altura da bola usando um controlador PID.*
- void Dados_recebidos (void)

  *Lê os dados recebidos pela UART e armazena-os no buffer para processamento.*
- void modo_valvula (void)

  *Controla a válvula para ajustar a posição da bola usando um controlador PI.*
- void Comando_PIeD (void)

  *Executa o controle PI ou PID, dependendo do modo de operação selecionado.*
- void inicia_ultrasom (void)

  *Inicia a medição da altura da bola utilizando um sensor ultrassônico.*
- void ProcessaDados (void)

  *Processa os dados recebidos após um intervalo de tempo.*
- int24_t map_value (int32_t x, int32_t in_min, int32_t in_max, int32_t out_min, int32_t out_max)

  *Converte um valor de um intervalo para outro.*

### 2.1.1 Descrição detalhada

Declaração das funções de controle, aquisição de dados e movimentação do motor.

Este arquivo contém os protótipos de funções responsáveis pelo controle do sistema, incluindo aquisição de dados do sensor ultrassônico, processamento da comunicação UART, e controle dos atuadores.

Autores: Herbert Tavares, Julio Cesar Carvalhes, Pedro Sampaio, Rafael Brasileiro

### 2.1.2 Documentação das funções

#### 2.1.2.1 Comando_PIeD()

```
void Comando_PIeD (
            void  )
```

Executa o controle PI ou PID, dependendo do modo de operação selecionado.

#### 2.1.2.2 Dados_recebidos()

```
void Dados_recebidos (
            void  )
```

Lê os dados recebidos pela UART e armazena-os no buffer para processamento.

#### 2.1.2.3 DRMotordePassos()

```
void DRMotordePassos ( )
```

Atualiza os passos do motor de acordo com a posição desejada.

#### 2.1.2.4 EnviaTx()

```
void EnviaTx (
            void  )
```

Carrega o buffer de transmissão (Tx) com as informações desejadas.

#### 2.1.2.5 funcao_aumento_num_passos()

```
void funcao_aumento_num_passos ( )
```

Realiza a mudança dos passos no sentido original.

#### 2.1.2.6 funcao_diminui_num_passos()

```
void funcao_diminui_num_passos ( )
```

Realiza a mudança dos passos no sentido contrário ao original.

#### 2.1.2.7 inicia_ultrasom()

```
void inicia_ultrasom (
            void  )
```

Inicia a medição da altura da bola utilizando um sensor ultrassônico.

#### 2.1.2.8 map_value()

```
int24_t map_value (
            int32_t x,
            int32_t in_min,
            int32_t in_max,
            int32_t out_min,
            int32_t out_max )
```

Converte um valor de um intervalo para outro.

Esta função mapeia o valor de 'x' do intervalo [in_min, in_max] para o intervalo [out_min, out_max].

**Parâmetros**

| | |
|---|---|
| *x* | Valor atual a ser mapeado. |
| *in_min* | Limite inferior do intervalo de entrada. |
| *in_max* | Limite superior do intervalo de entrada. |
| *out_min* | Limite inferior do intervalo de saída. |
| *out_max* | Limite superior do intervalo de saída. |

**Retorna**

Valor mapeado correspondente.

#### 2.1.2.9 media()

```
void media (
            uint16_t timer )
```

Calcula a média do sistema com base no tempo medido.

**Parâmetros**

| | |
|---|---|
| *timer* | Valor lido do timer. |

#### 2.1.2.10 medir_altura()

```
void medir_altura (
            void  )
```

Realiza a medição da altura.

#### 2.1.2.11 modo_valvula()

```
void modo_valvula (
            void  )
```

Controla a válvula para ajustar a posição da bola usando um controlador PI.

#### 2.1.2.12 modo_ventoinha()

```
void modo_ventoinha (
            void  )
```

Controla a ventoinha para ajustar a altura da bola usando um controlador PID.

#### 2.1.2.13 Passo1()

```
void Passo1 ( )
```

Aciona o primeiro passo do motor.

#### 2.1.2.14 Passo2()

```
void Passo2 ( )
```

Aciona o segundo passo do motor.

#### 2.1.2.15 Passo3()

```
void Passo3 ( )
```

Aciona o terceiro passo do motor.

#### 2.1.2.16 Passo4()

```
void Passo4 ( )
```

Aciona o quarto passo do motor.

#### 2.1.2.17 posicao_bola_tubo()

```
void posicao_bola_tubo (
            void  )
```

Calcula a posição da bola no tubo.

#### 2.1.2.18 Posicao_Bola_Tubo()

```
void Posicao_Bola_Tubo (
            void  )
```

Adquire a posição da bola no tubo.

#### 2.1.2.19 ProcessaDados()

```
void ProcessaDados (
            void  )
```

Processa os dados recebidos após um intervalo de tempo.

#### 2.1.2.20 selecao_do_modo()

```
void selecao_do_modo (
            void  )
```

Seleciona o modo de operação do sistema e configura os parâmetros correspondentes.

#### 2.1.2.21 velocidade_som()

```
uint16_t velocidade_som ( )
```

Retorna a velocidade do som de acordo com a temperatura.

Esta função utiliza a tabela armazenada na EEPROM para determinar a velocidade do som com base na temperatura lida.

**Retorna**

Velocidade do som correspondente à temperatura.

## 2.2 Referência ao ficheiro main.h

Define macros, variáveis globais e estruturas utilizadas no sistema.

## Macros

- #define numero_maximo_de_passos 460

  *Número máximo de passos da válvula (limite físico).*
- #define maximo_pwm 1023

  *Valor máximo do ciclo de trabalho do PWM (10 bits).*
- #define quadro_completo_ABCD 5

  *Número total de bytes esperados na comunicação UART.*
- #define COMANDO_MANUAL 0x00

  *Modo de operação: Controle manual (sem PID).*
- #define COMANDO_VENTOINHA 0x01

  *Modo de operação: Controle pela ventoinha (PID ativo).*
- #define COMANDO_VALVULA 0x02

  *Modo de operação: Controle pela válvula (PI ativo).*
- #define COMANDO_RESET 0x03

  *Comando para resetar o sistema.*
- #define SIMULACAO false

  *Macro para a simulação.*
- #define DEBUG false

  *Macro para a debug.*

## Variáveis

- __eeprom uint16_t relacao_som_temperatura [51]

  *Tabela armazenada na memória EEPROM que mapeia a temperatura (°C) à velocidade do som.*
- union {
  uint16_t Dado
  struct {
  uint8_t blsb
  uint8_t bmsb
  }
  } Set_altura

  *Union para manipulação de dados de 16 bits, permitindo acesso aos bytes MSB e LSB.*
- union {
  uint16_t Dado
  struct {
  uint8_t blsb
  uint8_t bmsb
  }
  } Set_valvula
- union {
  uint16_t Dado
  struct {
  uint8_t blsb
  uint8_t bmsb
  }
  } Set_ciclo
- int16_t tempo_s

  *Tempo de amostragem ou variável auxiliar de tempo.*
- int24_t erro_atual = 0

*Diferença entre a altura desejada e a altura da bola (erro de controle).*

- int24_t soma_anterior

  *Variável auxiliar para testes em simulação.*
- int24_t erro_anterior = 0

  *Variável auxiliar para armazenar o ciclo de trabalho anterior do PWM.*
- int24_t integrativo = 0

  *Acumulador do termo integral do controlador.*
- int24_t proporcional = 0

  *Componente proporcional do controle PID.*
- int24_t derivativo = 0

  *Componente derivativo do controle PID.*
- int24_t soma_PID = 0

  *Soma total dos termos (P, I e D, se aplicável) do controlador.*
- int8_t RxBuffer [5]

  *Buffer de recepção para armazenar os bytes recebidos via UART.*
- int24_t ki_ventoinha = 2

  *Ganho do termo integral da ventoinha (valor dividido por 1000).*
- int24_t kd_ventoinha = 10000

  *Ganho do termo derivativo da ventoinha.*
- int24_t kp_ventoinha = 1300

  *Ganho do termo proporcional da ventoinha.*
- int24_t ki_valvula = 1000

  *Ganho do termo integral da válvula.*
- int24_t kp_valvula = 2950

  *Ganho do termo proporcional da válvula.*
- int16_t tempo_medio = 0

  *Média dos últimos tempos medidos pelo sensor ultrassônico.*
- int24_t altura_bola = 0

  *Última altura medida da bola.*
- int16_t posicao_val_atual = 0

  *Posição atual da válvula.*
- uint16_t temperatura

  *Valor de temperatura lido pelo sensor.*
- int16_t posicao_val_futura = 0

  *Posição futura da válvula determinada pelo controle.*
- int16_t ciclo_util_futuro = 0

  *Próximo valor do ciclo de trabalho da ventoinha determinado pelo controle.*
- int8_t npassos = 0

  *Número de passos do motor de passos para controlar a válvula.*
- int8_t dados_de_envio [15]

  *Buffer de 15 bytes para envio de dados pela UART.*
- uint8_t modo

  *Armazena o modo de operação do sistema (Manual, Ventoinha, Válvula, Reset).*
- bool dado_atual = false

  *Flag que indica se os dados recebidos são atuais.*
- int8_t countRx = 0

  *Contador de bytes recebidos via UART.*

### 2.2.1 Descrição detalhada

Define macros, variáveis globais e estruturas utilizadas no sistema.

Este arquivo contém as definições de limites, códigos de comando, tabela de velocidade do som em função da temperatura e as variáveis globais empregadas nos algoritmos de controle.

Autores: Herbert Tavares, Julio Cesar Carvalhes, Pedro Sampaio, Rafael Brasileiro

### 2.2.2 Documentação das macros

#### 2.2.2.1 COMANDO_MANUAL

```
#define COMANDO_MANUAL 0x00
```

Modo de operação: Controle manual (sem PID).

#### 2.2.2.2 COMANDO_RESET

```
#define COMANDO_RESET 0x03
```

Comando para resetar o sistema.

#### 2.2.2.3 COMANDO_VALVULA

```
#define COMANDO_VALVULA 0x02
```

Modo de operação: Controle pela válvula (PI ativo).

#### 2.2.2.4 COMANDO_VENTOINHA

```
#define COMANDO_VENTOINHA 0x01
```

Modo de operação: Controle pela ventoinha (PID ativo).

#### 2.2.2.5 DEBUG

```
#define DEBUG false
```

Macro para a debug.

#### 2.2.2.6 maximo_pwm

```
#define maximo_pwm 1023
```

Valor máximo do ciclo de trabalho do PWM (10 bits).

#### 2.2.2.7 numero_maximo_de_passos

```
#define numero_maximo_de_passos 460
```

Número máximo de passos da válvula (limite físico).

#### 2.2.2.8 quadro_completo_ABCD

```
#define quadro_completo_ABCD 5
```

Número total de bytes esperados na comunicação UART.

#### 2.2.2.9 SIMULACAO

```
#define SIMULACAO false
```

Macro para a simulação.

### 2.2.3 Documentação das variáveis

#### 2.2.3.1 altura_bola

```
int24_t altura_bola = 0
```

Última altura medida da bola.

#### 2.2.3.2 blsb

```
uint8_t blsb
```

Byte menos significativo (LSB).

#### 2.2.3.3 bmsb

```
uint8_t bmsb
```

Byte mais significativo (MSB).

#### 2.2.3.4 ciclo_util_futuro

```
int16_t ciclo_util_futuro = 0
```

Próximo valor do ciclo de trabalho da ventoinha determinado pelo controle.

#### 2.2.3.5 countRx

```
int8_t countRx = 0
```

Contador de bytes recebidos via UART.

#### 2.2.3.6 Dado

```
uint16_t Dado
```

Valor de 16 bits completo.

#### 2.2.3.7 dado_atual

```
bool dado_atual = false
```

Flag que indica se os dados recebidos são atuais.

#### 2.2.3.8 dados_de_envio

```
int8_t dados_de_envio[15]
```

Buffer de 15 bytes para envio de dados pela UART.

#### 2.2.3.9 derivativo

```
int24_t derivativo = 0
```

Componente derivativo do controle PID.

#### 2.2.3.10 erro_anterior

```
int24_t erro_anterior = 0
```

Variável auxiliar para armazenar o ciclo de trabalho anterior do PWM.

#### 2.2.3.11 erro_atual

```
int24_t erro_atual = 0
```

Diferença entre a altura desejada e a altura da bola (erro de controle).

#### 2.2.3.12 integrativo

```
int24_t integrativo = 0
```

Acumulador do termo integral do controlador.

#### 2.2.3.13 kd_ventoinha

```
int24_t kd_ventoinha = 10000
```

Ganho do termo derivativo da ventoinha.

#### 2.2.3.14 ki_valvula

```
int24_t ki_valvula = 1000
```

Ganho do termo integral da válvula.

#### 2.2.3.15 ki_ventoinha

```
int24_t ki_ventoinha = 2
```

Ganho do termo integral da ventoinha (valor dividido por 1000).

#### 2.2.3.16 kp_valvula

```
int24_t kp_valvula = 2950
```

Ganho do termo proporcional da válvula.

#### 2.2.3.17 kp_ventoinha

```
int24_t kp_ventoinha = 1300
```

Ganho do termo proporcional da ventoinha.

#### 2.2.3.18 modo

```
uint8_t modo
```

Armazena o modo de operação do sistema (Manual, Ventoinha, Válvula, Reset).

#### 2.2.3.19 npassos

```
int8_t npassos = 0
```

Número de passos do motor de passos para controlar a válvula.

#### 2.2.3.20 posicao_val_atual

```
int16_t posicao_val_atual = 0
```

Posição atual da válvula.

#### 2.2.3.21 posicao_val_futura

```
int16_t posicao_val_futura = 0
```

Posição futura da válvula determinada pelo controle.

#### 2.2.3.22 proporcional

```
int24_t proporcional = 0
```

Componente proporcional do controle PID.

#### 2.2.3.23 relacao_som_temperatura

```
__eeprom uint16_t relacao_som_temperatura[51]
```

**Valor inicial:**

```
= {
    33145, 33206, 33266, 33327, 33387, 33447, 33507, 33567, 33627, 33687,
    33746, 33806, 33865, 33925, 33984, 34043, 34102, 34161, 34220, 34278,
    34337, 34396, 34454, 34512, 34570, 34629, 34687, 34745, 34802, 34860,
    34918, 34975, 35033, 35090, 35147, 35205, 35262, 35319, 35375, 35432,
    35489, 35546, 35602, 35659, 35715, 35771, 35827, 35883, 35939, 35995,
    36051
}
```

Tabela armazenada na memória EEPROM que mapeia a temperatura (°C) à velocidade do som.

Cada índice representa um valor de temperatura em °C e retorna a velocidade do som correspondente.

#### 2.2.3.24 RxBuffer

```
int8_t RxBuffer[5]
```

Buffer de recepção para armazenar os bytes recebidos via UART.

#### 2.2.3.25

```
union { ...  } Set_altura
```

Union para manipulação de dados de 16 bits, permitindo acesso aos bytes MSB e LSB.

Utilizada para transmissão e recepção de valores pela UART.

#### 2.2.3.26

```
union { ...  } Set_ciclo
```

#### 2.2.3.27

```
union { ...  } Set_valvula
```

#### 2.2.3.28 soma_anterior

```
int24_t soma_anterior
```

Variável auxiliar para testes em simulação.

#### 2.2.3.29 soma_PID

```
int24_t soma_PID = 0
```

Soma total dos termos (P, I e D, se aplicável) do controlador.

#### 2.2.3.30 temperatura

```
uint16_t temperatura
```

Valor de temperatura lido pelo sensor.

#### 2.2.3.31 tempo_medio

```
int16_t tempo_medio = 0
```

Média dos últimos tempos medidos pelo sensor ultrassônico.

#### 2.2.3.32 tempo_s

```
int16_t tempo_s
```

Tempo de amostragem ou variável auxiliar de tempo.