

Ricardo display 4 bit pic
2021.08

Gerado por Doxygen 1.8.17

| | |
|--|----------|
| 1 Freitas & Eletrônica | 1 |
| 2 Índice dos Arquivos | 3 |
| 2.1 Lista de Arquivos | 3 |
| 3 Arquivos | 5 |
| 3.1 Referência do Arquivo config.h | 5 |
| 3.1.1 Descrição detalhada | 6 |
| 3.2 Referência do Arquivo LCD.c | 6 |
| 3.2.1 Definições e macros | 7 |
| 3.2.1.1 M_LCD_SIZE | 7 |
| 3.2.2 Funções | 7 |
| 3.2.2.1 LCD_DATA() | 8 |
| 3.2.2.2 LCD_CMD_Nible() | 8 |
| 3.2.2.3 LCD_CMD_Word() | 9 |
| 3.2.2.4 LCD_Clear() | 10 |
| 3.2.2.5 LCD_Set_Cursor() | 10 |
| 3.2.2.6 LCD_Init() | 11 |
| 3.2.2.7 LCD_Write_Char() | 13 |
| 3.2.2.8 LCD_SL() | 14 |
| 3.2.2.9 LCD_SR() | 14 |
| 3.2.2.10 LCD_Remove_Char() | 15 |
| 3.2.2.11 LCD_Inserir_Char() | 16 |
| 3.2.2.12 LCD_Inserir_String() | 16 |
| 3.2.2.13 LCD_Has_Empty() | 17 |
| 3.2.2.14 LCD_MaqEstadoLiberar() | 17 |
| 3.2.2.15 LCD_MaqEstado() | 18 |
| 3.2.3 Variáveis | 19 |
| 3.2.3.1 M_LCD | 19 |
| 3.2.3.2 M_LCD_inserir | 20 |
| 3.2.3.3 M_LCD_remove | 20 |
| 3.2.3.4 ME_Running | 20 |
| 3.2.3.5 LCD_Current_Row | 20 |
| 3.2.3.6 LCD_Next_Col | 20 |
| 3.2.3.7 LCD_Number_Col | 20 |
| 3.2.3.8 LCD_Number_Row | 21 |
| 3.3 Referência do Arquivo LCD.h | 21 |
| 3.3.1 Descrição detalhada | 23 |
| 3.3.2 Definições e macros | 23 |
| 3.3.2.1 _XTAL_FREQ | 23 |
| 3.3.2.2 LCD_EN_Delay | 23 |
| 3.3.2.3 LCD_RS | 23 |
| 3.3.2.4 LCD_EN | 24 |

| | |
|-------------------------|----|
| 3.3.2.5 LCD_RW | 24 |
| 3.3.2.6 LCD_D4 | 24 |
| 3.3.2.7 LCD_D5 | 24 |
| 3.3.2.8 LCD_D6 | 24 |
| 3.3.2.9 LCD_D7 | 24 |
| 3.3.2.10 LCD_cmd_Clear | 25 |
| 3.3.2.11 LCD_cmd_R1 | 25 |
| 3.3.2.12 LCD_cmd_R2 | 25 |
| 3.3.2.13 LCD_cmd_R3 | 25 |
| 3.3.2.14 LCD_cmd_R4 | 25 |
| 3.3.2.15 LCD_cmd_SL | 25 |
| 3.3.2.16 LCD_cmd_SR | 26 |
| 3.3.2.17 LCD_cmd_2R | 26 |
| 3.3.2.18 LCD_cmd_4R | 26 |
| 3.3.2.19 LCD_cmd_16C | 26 |
| 3.3.2.20 LCD_cmd_20C | 26 |
| 3.3.2.21 LCD_Cursor_off | 26 |
| 3.3.2.22 LCD_Cursor_on | 27 |
| 3.3.2.23 LCD_Cursor_blk | 27 |
| 3.3.2.24 LCD_cmd_Col00 | 27 |
| 3.3.2.25 LCD_cmd_Col01 | 27 |
| 3.3.2.26 LCD_cmd_Col02 | 27 |
| 3.3.2.27 LCD_cmd_Col03 | 27 |
| 3.3.2.28 LCD_cmd_Col04 | 28 |
| 3.3.2.29 LCD_cmd_Col05 | 28 |
| 3.3.2.30 LCD_cmd_Col06 | 28 |
| 3.3.2.31 LCD_cmd_Col07 | 28 |
| 3.3.2.32 LCD_cmd_Col08 | 28 |
| 3.3.2.33 LCD_cmd_Col09 | 28 |
| 3.3.2.34 LCD_cmd_Col10 | 29 |
| 3.3.2.35 LCD_cmd_Col11 | 29 |
| 3.3.2.36 LCD_cmd_Col12 | 29 |
| 3.3.2.37 LCD_cmd_Col13 | 29 |
| 3.3.2.38 LCD_cmd_Col14 | 29 |
| 3.3.2.39 LCD_cmd_Col15 | 29 |
| 3.3.3 Funções | 30 |
| 3.3.3.1 LCD_Init() | 30 |
| 3.3.3.2 LCD_Clear() | 31 |
| 3.3.3.3 LCD_SL() | 32 |
| 3.3.3.4 LCD_SR() | 33 |
| 3.3.3.5 LCD_CMD_Nible() | 33 |
| 3.3.3.6 LCD_DATA() | 34 |

| | |
|-------------------------------------|-----------|
| 3.3.3.7 LCD_Set_Cursor() | 35 |
| 3.3.3.8 LCD_Write_Char() | 36 |
| 3.3.3.9 LCD_Inserir_String() | 37 |
| 3.3.3.10 LCD_Remove_Char() | 37 |
| 3.3.3.11 LCD_Inserir_Char() | 38 |
| 3.3.3.12 LCD_Has_Empty() | 38 |
| 3.3.3.13 LCD_MostrarEstado() | 39 |
| 3.3.3.14 LCD_MostrarEstadoLiberar() | 41 |
| 3.4 Referência do Arquivo main.c | 42 |
| 3.4.1 Definições e macros | 43 |
| 3.4.1.1 T0_cfg | 43 |
| 3.4.1.2 T1_cfg | 43 |
| 3.4.2 Funções | 43 |
| 3.4.2.1 ConfigureCPU() | 43 |
| 3.4.2.2 main() | 45 |
| 3.4.2.3 __interrupt() | 46 |
| 3.4.3 Variáveis | 47 |
| 3.4.3.1 LCD_Test_Char | 47 |
| Índice Remissivo | 49 |

Capítulo 1

Freitas & Eletrônica

Este projeto é composto de um LCD comandado pela comunicação serial.

Capítulo 2

Índice dos Arquivos

2.1 Lista de Arquivos

Esta é a lista de todos os arquivos e suas respectivas descrições:

| | | |
|--------------------------|---|----|
| config.h | PIC16F627A Configuration Bit Settings 'C' source line config statements pragma CONFIG | 5 |
| LCD.c | | 6 |
| LCD.h | Prototipos para LCD.c | 21 |
| main.c | | 42 |

Capítulo 3

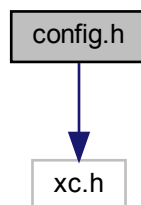
Arquivos

3.1 Referência do Arquivo config.h

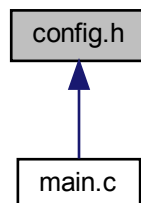
PIC16F627A Configuration Bit Settings
'C' source line config statements
pragma CONFIG.

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

Gráfico de dependência de inclusões para config.h:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



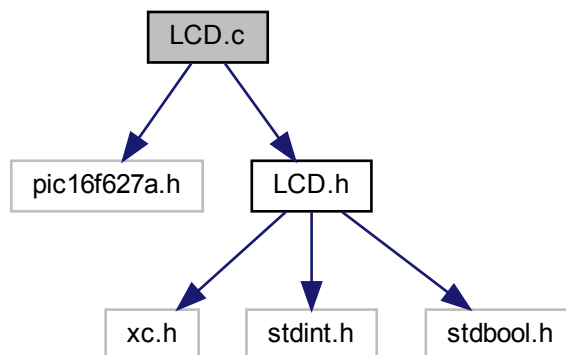
3.1.1 Descrição detalhada

PIC16F627A Configuration Bit Settings
'C' source line config statements
pragma CONFIG.

3.2 Referência do Arquivo LCD.c

```
#include <pic16f627a.h>  
#include "LCD.h"
```

Gráfico de dependência de inclusões para LCD.c:



Definições e Macros

- #define `M_LCD_SIZE` 0x4F

Funções

- void `LCD_DATA` (uint8_t Data)
Send Command To LCD.
- void `LCD_CMD_Nible` (uint8_t CMD)
Shift The Entire Display To The Right.
- void `LCD_CMD_Word` (uint8_t CMD)
- void `LCD_Clear` (void)
Initialize The LCD For 4-Bit Interface.
- void `LCD_Set_Cursor` (uint8_t row, uint8_t col)
Send 4-Bit Data To LCD.
- void `LCD_Init` (void)
- void `LCD_Write_Char` (char Data)
Set Cursor Position.
- void `LCD_SL` (void)

Clear The LCD Display.

- void `LCD_SR` (void)

Shift The Entire Display To The Left.

- char `LCD_Remove_Char` (void)

Write A String To LCD.

- void `LCD_Inserir_Char` (char dado)
- void `LCD_Inserir_String` (char *dado)

Write Character To LCD At Current Position.

- uint8_t `LCD_Has_Empty` (void)
- void `LCD_MaqEstadoLiberar` (void)

ME is able to run.

- void `LCD_MaqEstado` (void)

Variáveis

- char `M_LCD` [`M_LCD_SIZE`]
- uint8_t `M_LCD_inserir` = 0
- uint8_t `M_LCD_remove` = 0
- uint8_t `ME_Running`
- uint8_t `LCD_Current_Row` = 0x01
- uint8_t `LCD_Next_Col` = 0x00
- uint8_t `LCD_Number_Col` = 0x0F
- uint8_t `LCD_Number_Row` = 0x01

3.2.1 Definições e macros

3.2.1.1 M_LCD_SIZE

```
#define M_LCD_SIZE 0x4F
```

Alphanumeric LCD 16x2 or 16x4 4 bits of bus data/command

Definição na linha 13 do arquivo LCD.c.

3.2.2 Funções

3.2.2.1 LCD_DATA()

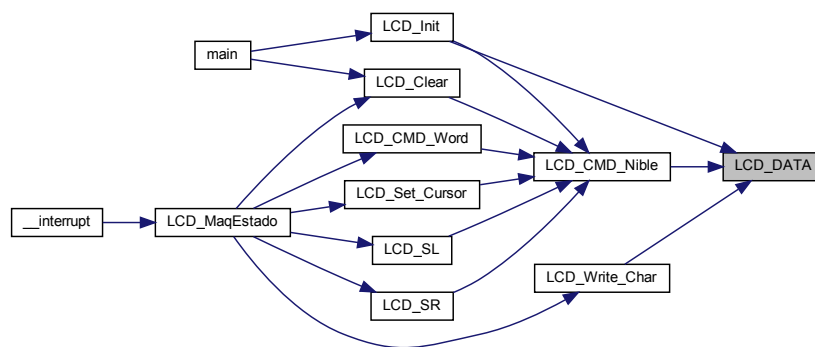
```
void LCD_DATA (
    uint8_t Data )
```

Send Command To LCD.

Definição na linha 29 do arquivo LCD.c.

```
29 {
30     LCD_D4 = (Data & 0x01) ? 1 : 0;
31     LCD_D5 = (Data & 0x02) ? 1 : 0;
32     LCD_D6 = (Data & 0x04) ? 1 : 0;
33     LCD_D7 = (Data & 0x08) ? 1 : 0;
34 }
```

Esse é o diagrama das funções que utilizam essa função:



3.2.2.2 LCD_CMD_Nibble()

```
void LCD_CMD_Nibble (
    uint8_t CMD )
```

Shift The Entire Display To The Right.

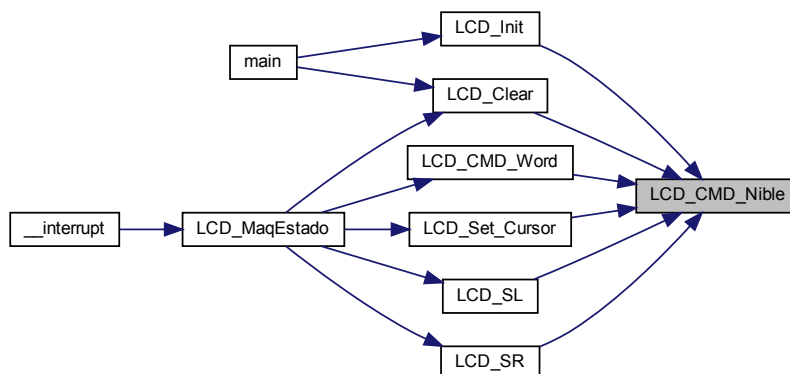
Definição na linha 40 do arquivo LCD.c.

```
40 {
41     LCD_RW = 0; // write info
42     LCD_RS = 0; // Select Command Register
43     LCD_DATA(CMD); // Move The Command Data To LCD
44     LCD_EN = 1; // Send The EN Clock Signal
45     _delay(LCD_EN_Delay);
46     LCD_EN = 0;
47 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



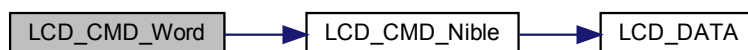
3.2.2.3 LCD_CMD_Word()

```
void LCD_CMD_Word (
    uint8_t CMD )
```

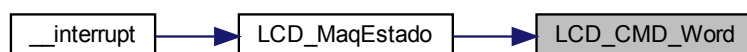
Definição na linha 53 do arquivo LCD.c.

```
53 {
54     uint8_t Low4, High4;
55     High4 = CMD >> 4;
56     Low4 = CMD & 0x0F;
57     LCD_CMD_Nibble(High4);
58     LCD_CMD_Nibble(Low4);
59 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.2.4 LCD_Clear()

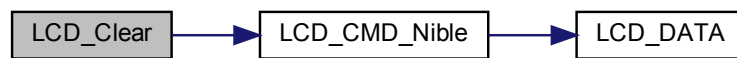
```
LCD_Clear (
    void )
```

Initialize The LCD For 4-Bit Interface.

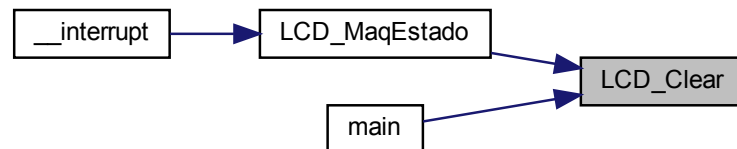
Definição na linha 65 do arquivo LCD.c.

```
65     {
66         LCD_CMD_Nibble(0);
67         LCD_CMD_Nibble(1);
68     }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.2.5 LCD_Set_Cursor()

```
void LCD_Set_Cursor (
    uint8_t row,
    uint8_t col )
```

Send 4-Bit Data To LCD.

Definição na linha 76 do arquivo LCD.c.

```
76     {
77         uint8_t Temp, Low4, High4;
78         if (row == 1)
79             Temp = 0x80 + col;
80
81         if (row == 2)
82             Temp = 0xC0 + col;
83     }
```

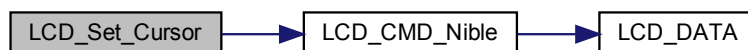


```

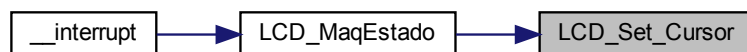
84     if (row == 3)
85         Temp = 0x80 + 0x10 + col;
86
87     if (row == 4)
88         Temp = 0xC0 + 0x10 + col;
89
90     LCD_Current_Row = row;
91     LCD_Next_Col = col + 1;
92     High4 = Temp >> 4;
93     Low4 = Temp & 0x0F;
94     LCD_CMD_Nibble(High4);
95     LCD_CMD_Nibble(Low4);
96 }

```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.2.6 LCD_Init()

```

LCD_Init (
    void )

```

LCD_MaqEstado não habilitada

0F = display on/off control

3F = set 8 bits 1/3

3F = set 8 bits2/3

3F = set 8 bits 3/3

2F = set 4 bits

28 : 2 line 5x7 dot

0C = cursor off 0E = cursor on 0F = blinking

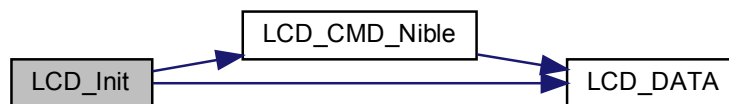
06 = Entry Mode

LCD_MaqEstado liberada

Definição na linha 101 do arquivo LCD.c.

```
101      {
102      ME_Running = 1; /** LCD_MaqEstado nãotilde;o habilitada */
103
104      // IO Pin Configurations in ConfigureCPU()
105      LCD_RW = 0; // write data/command
106      LCD_EN = 0;
107
108      _delay(30000); //Time before anything by Freitas
109      // The Init. Procedure As Described In The Datasheet
110      LCD_DATA(0x00); /** 0F = display on/off control */
111      _delay(30000);
112      _delay(LCD_EN_Delay);
113
114      LCD_CMD_Nible(0x03); /** 3F = set 8 bits 1/3 */
115      _delay(5000);
116
117      LCD_CMD_Nible(0x03); /** 3F = set 8 bits2/3 */
118      _delay(150);
119
120      LCD_CMD_Nible(0x03); /** 3F = set 8 bits 3/3 */
121
122      LCD_CMD_Nible(0x02); /** 2F = set 4 bits */
123
124      LCD_CMD_Nible(0x02); /** 28 : 2 line 5x7 dot*/
125      LCD_CMD_Nible(0x08);
126
127      /**
128      * 0C = cursor off
129      * 0E = cursor on
130      * 0F = blinking
131      */
132      LCD_CMD_Nible(0x00);
133      LCD_CMD_Nible(0x0C);
134
135      LCD_CMD_Nible(0x00); /** 06 = Entry Mode */
136      LCD_CMD_Nible(0x06);
137      ME_Running = 0; /** LCD_MaqEstado liberada */
138 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.2.7 LCD_Write_Char()

```
void LCD_Write_Char (
    char Data )
```

Set Cursor Position.

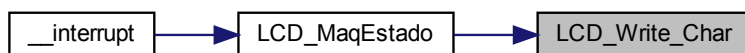
Definição na linha 144 do arquivo LCD.c.

```
144     {
145         uint8_t Low4, High4;
146
147         Low4 = Data & 0x0F;
148         //High4 = Data & 0xF0;
149         //High4 = High4 >> 4;
150         High4 = Data >> 4;
151
152         LCD_RS = 1;
153
154         LCD_DATA(High4);
155         LCD_EN = 1;
156         _delay(LCD_EN_Delay);
157         LCD_EN = 0;
158         _delay(LCD_EN_Delay);
159
160         LCD_DATA(Low4);
161         LCD_EN = 1;
162         _delay(LCD_EN_Delay);
163         LCD_EN = 0;
164         _delay(LCD_EN_Delay);
165         ++LCD_Next_Col;
166     }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.2.8 LCD_SL()

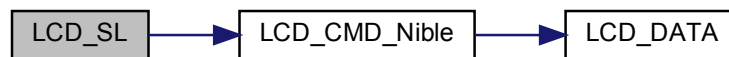
```
LCD_SL (
    void )
```

Clear The LCD Display.

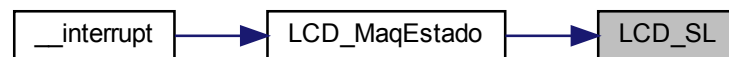
Definição na linha 171 do arquivo LCD.c.

```
171 {
172     LCD_CMD_Nibble(0x01);
173     LCD_CMD_Nibble(0x08);
174     --LCD_Next_Col;
175 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.2.9 LCD_SR()

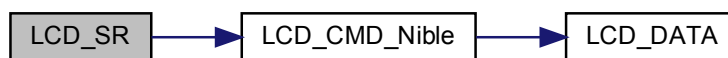
```
LCD_SR (
    void )
```

Shift The Entire Display To The Left.

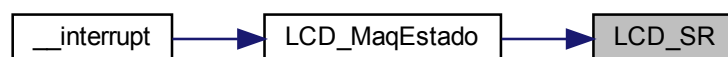
Definição na linha 180 do arquivo LCD.c.

```
180 {
181     LCD_CMD_Nibble(0x01);
182     LCD_CMD_Nibble(0x0C);
183     ++LCD_Next_Col;
184 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.2.10 LCD_Remove_Char()

```

LCD_Remove_Char (
    void )
  
```

Write A String To LCD.

Parâmetros

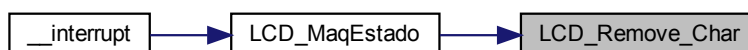
| | | |
|-----|------|--|
| out | char | |
|-----|------|--|

Definição na linha 190 do arquivo LCD.c.

```

190     {
191         M_LCD_remove++;
192         if (M_LCD_remove == M_LCD_SIZE) {
193             M_LCD_remove = 0;
194         }
195         return M_LCD[M_LCD_remove];
196     }
  
```

Esse é o diagrama das funções que utilizam essa função:



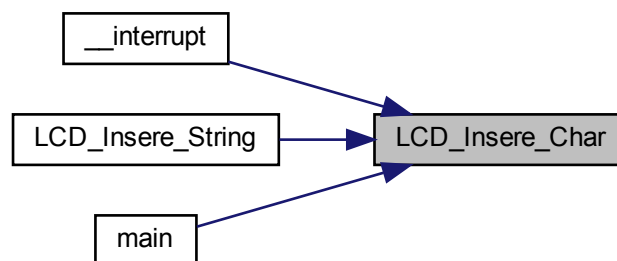
3.2.2.11 LCD_Inserere_Char()

```
void LCD_Inserere_Char (  
    char dado )
```

Definição na linha 202 do arquivo LCD.c.

```
202     {  
203         M_LCD_inserere++;  
204         if (M_LCD_inserere == M_LCD_SIZE) {  
205             M_LCD_inserere = 0;  
206         }  
207         M_LCD[M_LCD_inserere] = dado;  
208     }
```

Esse é o diagrama das funções que utilizam essa função:



3.2.2.12 LCD_Inserere_String()

```
void LCD_Inserere_String (  
    char * dado )
```

Write Character To LCD At Current Position.

Definição na linha 214 do arquivo LCD.c.

```
214     {  
215         int i;  
216         for (i = 0; dado[i] != '\0'; i++) {  
217             LCD_Inserere_Char(dado[i]);  
218         }  
219     }
```

Este é o diagrama das funções utilizadas por essa função:



3.2.2.13 LCD_Has_Empty()

```
LCD_Has_Empty (
    void )
```

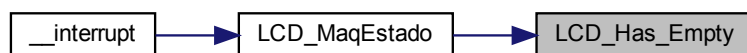
Parâmetros

| | | |
|-----|-----|----|
| out | yes | no |
|-----|-----|----|

Definição na linha 225 do arquivo LCD.c.

```
225     {
226     return (M_LCD_remove == M_LCD_inserere) ? 1 : 0;
227 }
```

Esse é o diagrama das funções que utilizam essa função:



3.2.2.14 LCD_MaqEstadoLiberar()

```
LCD_MaqEstadoLiberar (
    void )
```

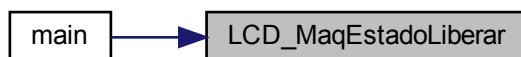
ME is able to run.

LCD_MaqEstado liberada

Definição na linha 233 do arquivo LCD.c.

```
233     {
234     ME_Running = false; /** LCD_MaqEstado liberada */
235 }
```

Esse é o diagrama das funções que utilizam essa função:



3.2.2.15 LCD_MaqEstado()

```
LCD_MaqEstado (
    void )
```

Definição na linha 240 do arquivo LCD.c.

```
240     {
241         uint8_t Col;
242
243         if (ME_Running) {
244             return;
245         }
246
247         if (LCD_Has_Empty()) {
248             ME_Running = false;
249             return;
250         } else {
251             ME_Running = true;
252         }
253
254         char c = LCD_Remove_Char();
255         switch (c) {
256             case LCD_cmd_Clear:
257                 LCD_Clear();
258                 break;
259             case LCD_cmd_R1:
260                 LCD_Set_Cursor(0x01, 0x00);
261                 break;
262             case LCD_cmd_R2:
263                 LCD_Set_Cursor(0x02, 0x00);
264                 break;
265             case LCD_cmd_R3:
266                 LCD_Set_Cursor(0x03, 0x00);
267                 break;
268             case LCD_cmd_R4:
269                 LCD_Set_Cursor(0x04, 0x00);
270                 break;
271             case LCD_cmd_SL:
272                 LCD_SL();
273                 break;
274             case LCD_cmd_SR:
275                 LCD_SR();
276                 break;
277
278             case LCD_Cursor_off:
279                 LCD_CMD_Word(0x0C);
280                 break;
281             case LCD_Cursor_on:
282                 LCD_CMD_Word(0x0E);
283                 break;
284             case LCD_Cursor_blk:
285                 LCD_CMD_Word(0x0F);
286                 break;
287
288             case LCD_cmd_2R:
289                 LCD_Number_Row = 0x01;
290                 break;
291             case LCD_cmd_4R:
292                 LCD_Number_Row = 0x03;
293                 break;
294             case LCD_cmd_16C:
295                 LCD_Number_Col = 0x0F;
296                 break;
297             case LCD_cmd_20C:
298                 LCD_Number_Col = 0x13;
299                 break;
300
301             case LCD_cmd_Col00:
302             case LCD_cmd_Col01:
303             case LCD_cmd_Col02:
304             case LCD_cmd_Col03:
305             case LCD_cmd_Col04:
306             case LCD_cmd_Col05:
307             case LCD_cmd_Col06:
308             case LCD_cmd_Col07:
309             case LCD_cmd_Col08:
310             case LCD_cmd_Col09:
311             case LCD_cmd_Col10:
312             case LCD_cmd_Col11:
313             case LCD_cmd_Col12:
314             case LCD_cmd_Col13:
315             case LCD_cmd_Col14:
316             case LCD_cmd_Col15:
317                 Col = (c & 0x0F);
```

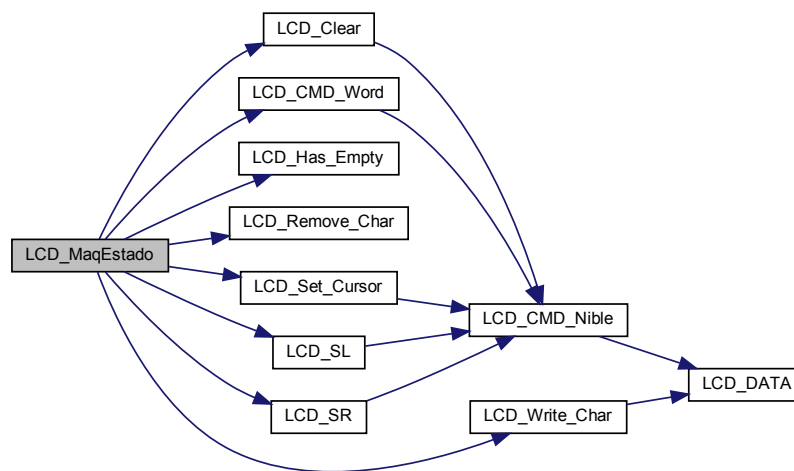


```

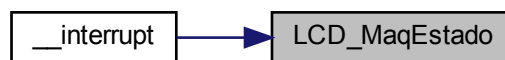
318         LCD_Set_Cursor(LCD_Current_Row, Col);
319         break;
320
321     default:
322         LCD_Write_Char(c);
323         if (LCD_Next_Col > 0x10) {
324             if (LCD_Current_Row < 4) {
325                 ++LCD_Current_Row;
326             } else {
327                 LCD_Current_Row = 1;
328             }
329             LCD_Set_Cursor(LCD_Current_Row, 0x00);
330         }
331         break;
332     }
333
334     ME_Running = false;
335 }

```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.2.3 Variáveis

3.2.3.1 M_LCD

```
char M_LCD[M_LCD_SIZE]
```

Definição na linha 15 do arquivo LCD.c.

3.2.3.2 M_LCD_inserere

```
uint8_t M_LCD_inserere = 0
```

Definição na linha 16 do arquivo LCD.c.

3.2.3.3 M_LCD_remove

```
uint8_t M_LCD_remove = 0
```

Definição na linha 17 do arquivo LCD.c.

3.2.3.4 ME_Running

```
uint8_t ME_Running
```

Definição na linha 18 do arquivo LCD.c.

3.2.3.5 LCD_Current_Row

```
uint8_t LCD_Current_Row = 0x01
```

Definição na linha 19 do arquivo LCD.c.

3.2.3.6 LCD_Next_Col

```
uint8_t LCD_Next_Col = 0x00
```

Definição na linha 20 do arquivo LCD.c.

3.2.3.7 LCD_Number_Col

```
uint8_t LCD_Number_Col = 0x0F
```

Definição na linha 22 do arquivo LCD.c.

3.2.3.8 LCD_Number_Row

```
uint8_t LCD_Number_Row = 0x01
```

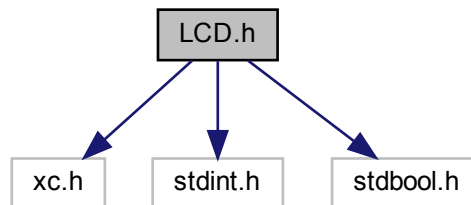
Definição na linha 23 do arquivo LCD.c.

3.3 Referência do Arquivo LCD.h

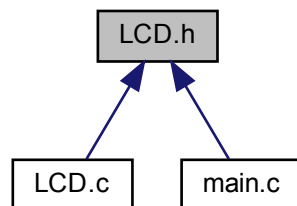
protótipos para [LCD.c](#)

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

Gráfico de dependência de inclusões para LCD.h:



Este grafo mostra quais arquivos estão direta ou indiretamente relacionados com esse arquivo:



Definições e Macros

- `#define _XTAL_FREQ 4000000`
For uint8_t definition.
- `#define LCD_EN_Delay 500`

- #define LCD_RS
- #define LCD_EN PORTAbits.RA1 /** @brief PIC(18) LCD Enable (6) */
- #define LCD_RW PORTAbits.RA2 /** @brief PIC(01) LCD RW (5) */
- #define LCD_D4 PORTBbits.RB4 /** @brief PIC(10) LCD Data 4 (11) */
- #define LCD_D5 PORTBbits.RB5 /** @brief PIC(11) LCD Data 5 (12) */
- #define LCD_D6 PORTBbits.RB6 /** @brief PIC(12) LCD Data 6 (13) */
- #define LCD_D7 PORTBbits.RB7 /** @brief PIC(13) LCD Data 7 (14) */
- #define LCD_cmd_Clear 0x01 /** @brief LCD_Clear() */
- #define LCD_cmd_R1 0x02 /** @brief LCD_Set_Cursor(0x01, 0x00) */
- #define LCD_cmd_R2 0x03 /** @brief LCD_Set_Cursor(0x02, 0x00) */
- #define LCD_cmd_R3 0x04 /** @brief LCD_Set_Cursor(0x03, 0x00) */
- #define LCD_cmd_R4 0x05 /** @brief LCD_Set_Cursor(0x04, 0x00) */
- #define LCD_cmd_SL 0x06 /** @brief LCD_Shift_Left */
- #define LCD_cmd_SR 0x07 /** @brief LCD_Shift_Right */
- #define LCD_cmd_2R 0x08 /** @brief LCD_Shift_Right */
- #define LCD_cmd_4R 0x09 /** @brief LCD_Shift_Right */
- #define LCD_cmd_16C 0x0A /** @brief LCD_Shift_Right */
- #define LCD_cmd_20C 0x0B /** @brief LCD_Shift_Right */
- #define LCD_Cursor_off 0x0C /** cursor off */
- #define LCD_Cursor_on 0x0E /** cursor on */
- #define LCD_Cursor_blink 0x0F /** blinking */
- #define LCD_cmd_Col00 0x10 /** @brief LCD_Set_Cursor to nn Col */
- #define LCD_cmd_Col01 0x11
- #define LCD_cmd_Col02 0x12
- #define LCD_cmd_Col03 0x13
- #define LCD_cmd_Col04 0x14
- #define LCD_cmd_Col05 0x15
- #define LCD_cmd_Col06 0x16
- #define LCD_cmd_Col07 0x17
- #define LCD_cmd_Col08 0x18
- #define LCD_cmd_Col09 0x19
- #define LCD_cmd_Col10 0x1A
- #define LCD_cmd_Col11 0x1B
- #define LCD_cmd_Col12 0x1C
- #define LCD_cmd_Col13 0x1D
- #define LCD_cmd_Col14 0x1E
- #define LCD_cmd_Col15 0x1F /** @brief LCD_Set_Cursor to nn Col */

Funções

- void LCD_Init (void)
- void LCD_Clear (void)
Initialize The LCD For 4-Bit Interface.
- void LCD_SL (void)
Clear The LCD Display.
- void LCD_SR (void)
Shift The Entire Display To The Left.
- void LCD_CMD_Nible (uint8_t)
Shift The Entire Display To The Right.
- void LCD_DATA (uint8_t)
Send Command To LCD.
- void LCD_Set_Cursor (uint8_t, uint8_t)
Send 4-Bit Data To LCD.

- void `LCD_Write_Char` (char)
Set Cursor Position.
- void `LCD_Inserir_String` (char *)
Write Character To LCD At Current Position.
- char `LCD_Remove_Char` (void)
Write A String To LCD.
- void `LCD_Inserir_Char` (char)
- uint8_t `LCD_Has_Empty` (void)
- void `LCD_MaqEstado` (void)
- void `LCD_MaqEstadoLiberar` (void)
ME is able to run.

3.3.1 Descrição detalhada

protótipos para `LCD.c`

3.3.2 Definições e macros

3.3.2.1 `_XTAL_FREQ`

```
#define _XTAL_FREQ 4000000
```

For uint8_t definition.

For true/false definition

Definição na linha 13 do arquivo LCD.h.

3.3.2.2 `LCD_EN_Delay`

```
#define LCD_EN_Delay 500
```

Definição na linha 14 do arquivo LCD.h.

3.3.2.3 `LCD_RS`

```
#define LCD_RS
```

Valor:

```
PORTAbits.RA0 /** @brief LCD RS (04) PIC(17)
 *             H-data input
 *             L-instruction input
 **/
```

Definição na linha 16 do arquivo LCD.h.

3.3.2.4 LCD_EN

```
#define LCD_EN PORTAbits.RA1 /** @brief PIC(18) LCD Enable (6) */
```

Definição na linha 17 do arquivo LCD.h.

3.3.2.5 LCD_RW

```
#define LCD_RW PORTAbits.RA2 /** @brief PIC(01) LCD RW (5) */
```

Definição na linha 18 do arquivo LCD.h.

3.3.2.6 LCD_D4

```
#define LCD_D4 PORTBbits.RB4 /** @brief PIC(10) LCD Data 4 (11) */
```

Definição na linha 23 do arquivo LCD.h.

3.3.2.7 LCD_D5

```
#define LCD_D5 PORTBbits.RB5 /** @brief PIC(11) LCD Data 5 (12) */
```

Definição na linha 24 do arquivo LCD.h.

3.3.2.8 LCD_D6

```
#define LCD_D6 PORTBbits.RB6 /** @brief PIC(12) LCD Data 6 (13) */
```

Definição na linha 25 do arquivo LCD.h.

3.3.2.9 LCD_D7

```
#define LCD_D7 PORTBbits.RB7 /** @brief PIC(13) LCD Data 7 (14) */
```

Definição na linha 26 do arquivo LCD.h.

3.3.2.10 LCD_cmd_Clear

```
#define LCD_cmd_Clear 0x01 /** @brief LCD_Clear() */
```

LCD area reservada para comandos especiais jump 00 ~ 1F jump 80 ~ 9F jump A6 ~ DA jump F9 ~ xx

Definição na linha 36 do arquivo LCD.h.

3.3.2.11 LCD_cmd_R1

```
#define LCD_cmd_R1 0x02 /** @brief LCD_Set_Cursor(0x01, 0x00) */
```

Definição na linha 37 do arquivo LCD.h.

3.3.2.12 LCD_cmd_R2

```
#define LCD_cmd_R2 0x03 /** @brief LCD_Set_Cursor(0x02, 0x00) */
```

Definição na linha 38 do arquivo LCD.h.

3.3.2.13 LCD_cmd_R3

```
#define LCD_cmd_R3 0x04 /** @brief LCD_Set_Cursor(0x03, 0x00) */
```

Definição na linha 39 do arquivo LCD.h.

3.3.2.14 LCD_cmd_R4

```
#define LCD_cmd_R4 0x05 /** @brief LCD_Set_Cursor(0x04, 0x00) */
```

Definição na linha 40 do arquivo LCD.h.

3.3.2.15 LCD_cmd_SL

```
#define LCD_cmd_SL 0x06 /** @brief LCD_Shift_Left */
```

Definição na linha 41 do arquivo LCD.h.

3.3.2.16 LCD_cmd_SR

```
#define LCD_cmd_SR 0x07 /** @brief LCD_Shift_Right */
```

Definição na linha 42 do arquivo LCD.h.

3.3.2.17 LCD_cmd_2R

```
#define LCD_cmd_2R 0x08 /** @brief LCD_Shift_Right */
```

Definição na linha 44 do arquivo LCD.h.

3.3.2.18 LCD_cmd_4R

```
#define LCD_cmd_4R 0x09 /** @brief LCD_Shift_Right */
```

Definição na linha 45 do arquivo LCD.h.

3.3.2.19 LCD_cmd_16C

```
#define LCD_cmd_16C 0x0A /** @brief LCD_Shift_Right */
```

Definição na linha 46 do arquivo LCD.h.

3.3.2.20 LCD_cmd_20C

```
#define LCD_cmd_20C 0x0B /** @brief LCD_Shift_Right */
```

Definição na linha 47 do arquivo LCD.h.

3.3.2.21 LCD_Cursor_off

```
#define LCD_Cursor_off 0x0C /** cursor off */
```

Definição na linha 49 do arquivo LCD.h.

3.3.2.22 LCD_Cursor_on

```
#define LCD_Cursor_on 0x0E /** cursor on */
```

Definição na linha 50 do arquivo LCD.h.

3.3.2.23 LCD_Cursor_blk

```
#define LCD_Cursor_blk 0x0F /** blinking */
```

Definição na linha 51 do arquivo LCD.h.

3.3.2.24 LCD_cmd_Col00

```
#define LCD_cmd_Col00 0x10 /** @brief LCD_Set_Cursor to nn Col */
```

Definição na linha 53 do arquivo LCD.h.

3.3.2.25 LCD_cmd_Col01

```
#define LCD_cmd_Col01 0x11
```

Definição na linha 54 do arquivo LCD.h.

3.3.2.26 LCD_cmd_Col02

```
#define LCD_cmd_Col02 0x12
```

Definição na linha 55 do arquivo LCD.h.

3.3.2.27 LCD_cmd_Col03

```
#define LCD_cmd_Col03 0x13
```

Definição na linha 56 do arquivo LCD.h.

3.3.2.28 LCD_cmd_Col04

```
#define LCD_cmd_Col04 0x14
```

Definição na linha 57 do arquivo LCD.h.

3.3.2.29 LCD_cmd_Col05

```
#define LCD_cmd_Col05 0x15
```

Definição na linha 58 do arquivo LCD.h.

3.3.2.30 LCD_cmd_Col06

```
#define LCD_cmd_Col06 0x16
```

Definição na linha 59 do arquivo LCD.h.

3.3.2.31 LCD_cmd_Col07

```
#define LCD_cmd_Col07 0x17
```

Definição na linha 60 do arquivo LCD.h.

3.3.2.32 LCD_cmd_Col08

```
#define LCD_cmd_Col08 0x18
```

Definição na linha 61 do arquivo LCD.h.

3.3.2.33 LCD_cmd_Col09

```
#define LCD_cmd_Col09 0x19
```

Definição na linha 62 do arquivo LCD.h.

3.3.2.34 LCD_cmd_Col10

```
#define LCD_cmd_Col10 0x1A
```

Definição na linha 63 do arquivo LCD.h.

3.3.2.35 LCD_cmd_Col11

```
#define LCD_cmd_Col11 0x1B
```

Definição na linha 64 do arquivo LCD.h.

3.3.2.36 LCD_cmd_Col12

```
#define LCD_cmd_Col12 0x1C
```

Definição na linha 65 do arquivo LCD.h.

3.3.2.37 LCD_cmd_Col13

```
#define LCD_cmd_Col13 0x1D
```

Definição na linha 66 do arquivo LCD.h.

3.3.2.38 LCD_cmd_Col14

```
#define LCD_cmd_Col14 0x1E
```

Definição na linha 67 do arquivo LCD.h.

3.3.2.39 LCD_cmd_Col15

```
#define LCD_cmd_Col15 0x1F /** @brief LCD_Set_Cursor to nn Col */
```

Definição na linha 68 do arquivo LCD.h.

3.3.3 Funções

3.3.3.1 LCD_Init()

```
void LCD_Init (
    void )
```

LCD_MaqEstado não habilitada

0F = display on/off control

3F = set 8 bits 1/3

3F = set 8 bits2/3

3F = set 8 bits 3/3

2F = set 4 bits

28 : 2 line 5x7 dot

0C = cursor off 0E = cursor on 0F = blinking

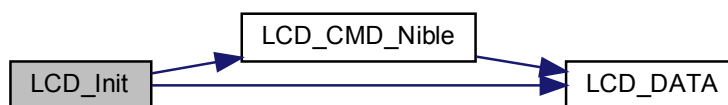
06 = Entry Mode

LCD_MaqEstado liberada

Definição na linha 101 do arquivo LCD.c.

```
101 {
102     ME_Running = 1; /** LCD_MaqEstado não habilitada */
103
104     // IO Pin Configurations in ConfigureCPU()
105     LCD_RW = 0; // write data/command
106     LCD_EN = 0;
107
108     _delay(30000); //Time before anything by Freitas
109     // The Init. Procedure As Described In The Datasheet
110     LCD_DATA(0x00); /** 0F = display on/off control */
111     _delay(30000);
112     _delay(LCD_EN_Delay);
113
114     LCD_CMD_Nibble(0x03); /** 3F = set 8 bits 1/3 */
115     _delay(5000);
116
117     LCD_CMD_Nibble(0x03); /** 3F = set 8 bits2/3 */
118     _delay(150);
119
120     LCD_CMD_Nibble(0x03); /** 3F = set 8 bits 3/3 */
121
122     LCD_CMD_Nibble(0x02); /** 2F = set 4 bits */
123
124     LCD_CMD_Nibble(0x02); /** 28 : 2 line 5x7 dot*/
125     LCD_CMD_Nibble(0x08);
126
127     /**
128     * 0C = cursor off
129     * 0E = cursor on
130     * 0F = blinking
131     */
132     LCD_CMD_Nibble(0x00);
133     LCD_CMD_Nibble(0x0C);
134
135     LCD_CMD_Nibble(0x00); /** 06 = Entry Mode */
136     LCD_CMD_Nibble(0x06);
137     ME_Running = 0; /** LCD_MaqEstado liberada */
138 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.2 LCD_Clear()

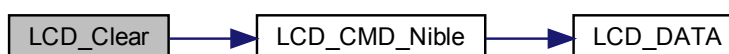
```
void LCD_Clear (
    void )
```

Initialize The LCD For 4-Bit Interface.

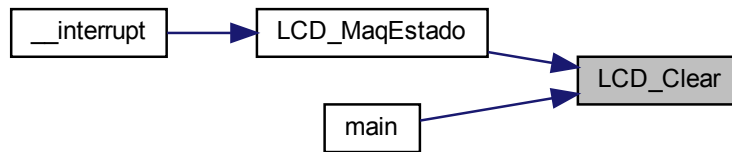
Definição na linha 65 do arquivo LCD.c.

```
65 {
66     LCD_CMD_Nibble(0);
67     LCD_CMD_Nibble(1);
68 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.3 LCD_SL()

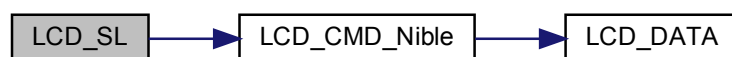
```
void LCD_SL (
    void )
```

Clear The LCD Display.

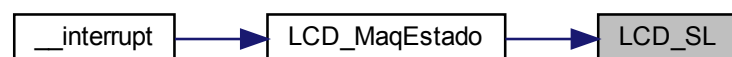
Definição na linha 171 do arquivo LCD.c.

```
171 {
172     LCD_CMD_Nibble (0x01);
173     LCD_CMD_Nibble (0x08);
174     --LCD_Next_Col;
175 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.4 LCD_SR()

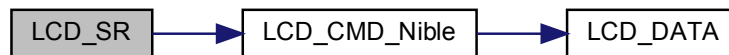
```
void LCD_SR (
    void )
```

Shift The Entire Display To The Left.

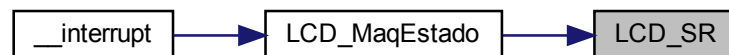
Definição na linha 180 do arquivo LCD.c.

```
180     {
181         LCD_CMD_Nible(0x01);
182         LCD_CMD_Nible(0x0C);
183         ++LCD_Next_Col;
184     }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.5 LCD_CMD_Nible()

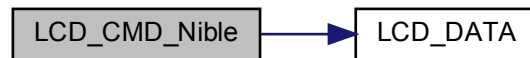
```
void LCD_CMD_Nible (
    uint8_t )
```

Shift The Entire Display To The Right.

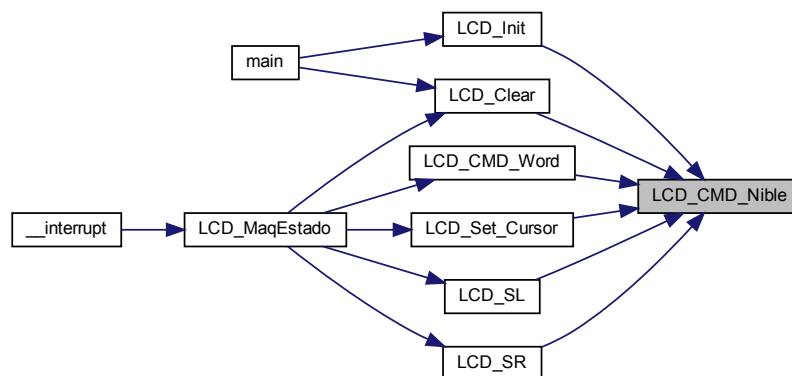
Definição na linha 40 do arquivo LCD.c.

```
40     {
41         LCD_RW = 0; // write info
42         LCD_RS = 0; // Select Command Register
43         LCD_DATA(CMD); // Move The Command Data To LCD
44         LCD_EN = 1; // Send The EN Clock Signal
45         _delay(LCD_EN_Delay);
46         LCD_EN = 0;
47     }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.6 LCD_DATA()

```
void LCD_DATA (
    uint8_t )
```

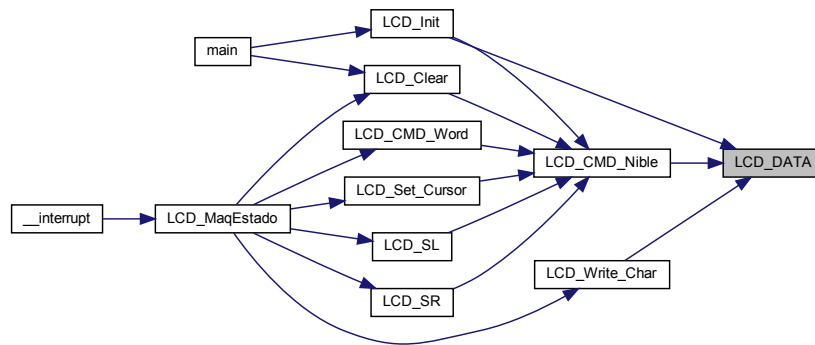
Send Command To LCD.

Definição na linha 29 do arquivo LCD.c.

```

29      {
30      LCD_D4 = (Data & 0x01) ? 1 : 0;
31      LCD_D5 = (Data & 0x02) ? 1 : 0;
32      LCD_D6 = (Data & 0x04) ? 1 : 0;
33      LCD_D7 = (Data & 0x08) ? 1 : 0;
34  }
```


Esse é o diagrama das funções que utilizam essa função:



3.3.3.7 LCD_Set_Cursor()

```
void LCD_Set_Cursor (
    uint8_t ,
    uint8_t )
```

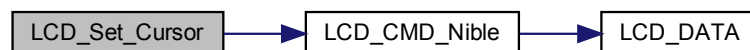
Send 4-Bit Data To LCD.

Definição na linha 76 do arquivo LCD.c.

```

76 {
77     uint8_t Temp, Low4, High4;
78     if (row == 1)
79         Temp = 0x80 + col;
80
81     if (row == 2)
82         Temp = 0xC0 + col;
83
84     if (row == 3)
85         Temp = 0x80 + 0x10 + col;
86
87     if (row == 4)
88         Temp = 0xC0 + 0x10 + col;
89
90     LCD_Current_Row = row;
91     LCD_Next_Col = col + 1;
92     High4 = Temp » 4;
93     Low4 = Temp & 0x0F;
94     LCD_CMD_Nibble(High4);
95     LCD_CMD_Nibble(Low4);
96 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.8 LCD_Write_Char()

```
void LCD_Write_Char (
    char )
```

Set Cursor Position.

Definição na linha 144 do arquivo LCD.c.

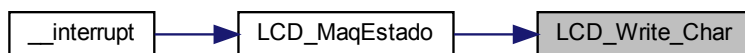
```

144     {
145         uint8_t Low4, High4;
146
147         Low4 = Data & 0x0F;
148         //High4 = Data & 0xF0;
149         //High4 = High4 >> 4;
150         High4 = Data >> 4;
151
152         LCD_RS = 1;
153
154         LCD_DATA(High4);
155         LCD_EN = 1;
156         _delay(LCD_EN_Delay);
157         LCD_EN = 0;
158         _delay(LCD_EN_Delay);
159
160         LCD_DATA(Low4);
161         LCD_EN = 1;
162         _delay(LCD_EN_Delay);
163         LCD_EN = 0;
164         _delay(LCD_EN_Delay);
165         ++LCD_Next_Col;
166     }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.9 LCD_Inserer_String()

```
void LCD_Inserer_String (
    char * )
```

Write Character To LCD At Current Position.

Definição na linha 214 do arquivo LCD.c.

```
214         {
215     int i;
216     for (i = 0; dado[i] != '\0'; i++) {
217         LCD_Inserer_Char(dado[i]);
218     }
219 }
```

Este é o diagrama das funções utilizadas por essa função:



3.3.3.10 LCD_Remove_Char()

```
char LCD_Remove_Char (
    void )
```

Write A String To LCD.

Parâmetros

| | | |
|-----|------|--|
| out | char | |
|-----|------|--|

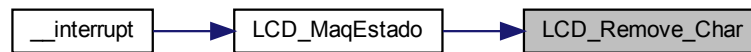
Definição na linha 190 do arquivo LCD.c.

```

190     {
191         M_LCD_remove++;
192         if (M_LCD_remove == M_LCD_SIZE) {
193             M_LCD_remove = 0;
194         }
195         return M_LCD[M_LCD_remove];
196     }

```

Esse é o diagrama das funções que utilizam essa função:



3.3.3.11 LCD_Inserere_Char()

```

void LCD_Inserere_Char (
    char )

```

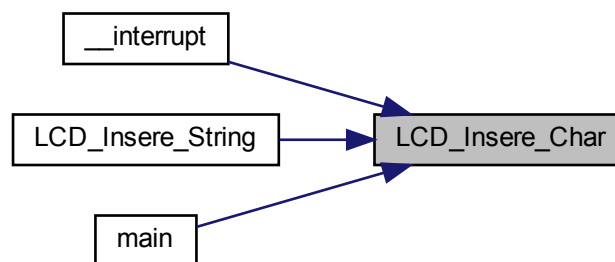
Definição na linha 202 do arquivo LCD.c.

```

202     {
203         M_LCD_inserere++;
204         if (M_LCD_inserere == M_LCD_SIZE) {
205             M_LCD_inserere = 0;
206         }
207         M_LCD[M_LCD_inserere] = dado;
208     }

```

Esse é o diagrama das funções que utilizam essa função:



3.3.3.12 LCD_Has_Empty()

```

uint8_t LCD_Has_Empty (
    void )

```

Parâmetros

| | | |
|-----|-----|----|
| out | yes | no |
|-----|-----|----|

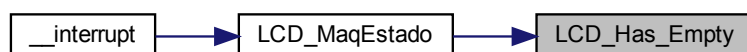
Definição na linha 225 do arquivo LCD.c.

```

225     {
226     return (M_LCD_remove == M_LCD_inserere) ? 1 : 0;
227 }

```

Esse é o diagrama das funções que utilizam essa função:



3.3.3.13 LCD_MaqEstado()

```

void LCD_MaqEstado (
    void )

```

Definição na linha 240 do arquivo LCD.c.

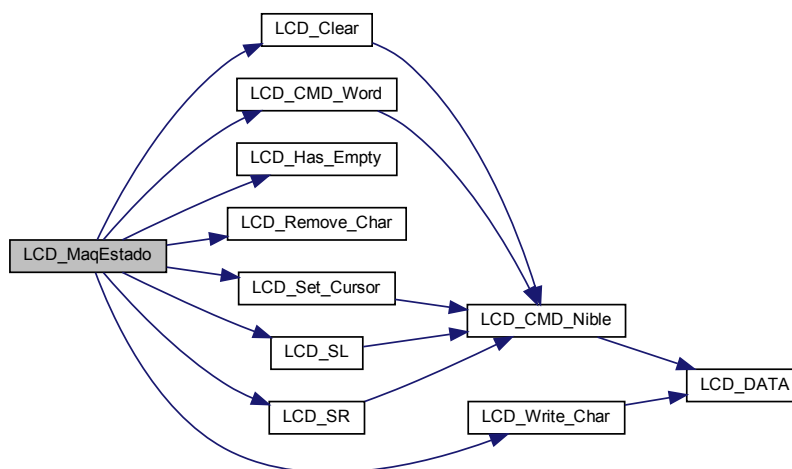
```

240     {
241     uint8_t Col;
242
243     if (ME_Running) {
244         return;
245     }
246
247     if (LCD_Has_Empty()) {
248         ME_Running = false;
249         return;
250     } else {
251         ME_Running = true;
252     }
253
254     char c = LCD_Remove_Char();
255     switch (c) {
256     case LCD_cmd_Clear:
257         LCD_Clear();
258         break;
259     case LCD_cmd_R1:
260         LCD_Set_Cursor(0x01, 0x00);
261         break;
262     case LCD_cmd_R2:
263         LCD_Set_Cursor(0x02, 0x00);
264         break;
265     case LCD_cmd_R3:
266         LCD_Set_Cursor(0x03, 0x00);
267         break;
268     case LCD_cmd_R4:
269         LCD_Set_Cursor(0x04, 0x00);
270         break;
271     case LCD_cmd_SL:
272         LCD_SL();
273         break;
274     case LCD_cmd_SR:
275         LCD_SR();
276         break;
277
278     case LCD_Cursor_off:
279         LCD_CMD_Word(0x0C);
280         break;

```

```
281     case LCD_Cursor_on:
282         LCD_CMD_Word(0x0E);
283         break;
284     case LCD_Cursor_blk:
285         LCD_CMD_Word(0x0F);
286         break;
287
288     case LCD_cmd_2R:
289         LCD_Number_Row = 0x01;
290         break;
291     case LCD_cmd_4R:
292         LCD_Number_Row = 0x03;
293         break;
294     case LCD_cmd_16C:
295         LCD_Number_Col = 0x0F;
296         break;
297     case LCD_cmd_20C:
298         LCD_Number_Col = 0x13;
299         break;
300
301     case LCD_cmd_Col00:
302     case LCD_cmd_Col01:
303     case LCD_cmd_Col02:
304     case LCD_cmd_Col03:
305     case LCD_cmd_Col04:
306     case LCD_cmd_Col05:
307     case LCD_cmd_Col06:
308     case LCD_cmd_Col07:
309     case LCD_cmd_Col08:
310     case LCD_cmd_Col09:
311     case LCD_cmd_Col10:
312     case LCD_cmd_Col11:
313     case LCD_cmd_Col12:
314     case LCD_cmd_Col13:
315     case LCD_cmd_Col14:
316     case LCD_cmd_Col15:
317         Col = (c & 0x0F);
318         LCD_Set_Cursor(LCD_Current_Row, Col);
319         break;
320
321     default:
322         LCD_Write_Char(c);
323         if (LCD_Next_Col > 0x10) {
324             if (LCD_Current_Row < 4) {
325                 ++LCD_Current_Row;
326             } else {
327                 LCD_Current_Row = 1;
328             }
329             LCD_Set_Cursor(LCD_Current_Row, 0x00);
330         }
331         break;
332 }
333
334 ME_Running = false;
335 }
```

Este é o diagrama das funções utilizadas por essa função:



Esse é o diagrama das funções que utilizam essa função:



3.3.3.14 LCD_MaqEstadoLiberar()

```
void LCD_MaqEstadoLiberar (
    void )
```

ME is able to run.

LCD_MaqEstado liberada

Definição na linha 233 do arquivo LCD.c.

```
233 {
234     ME_Running = false; /** LCD_MaqEstado liberada */
235 }
```

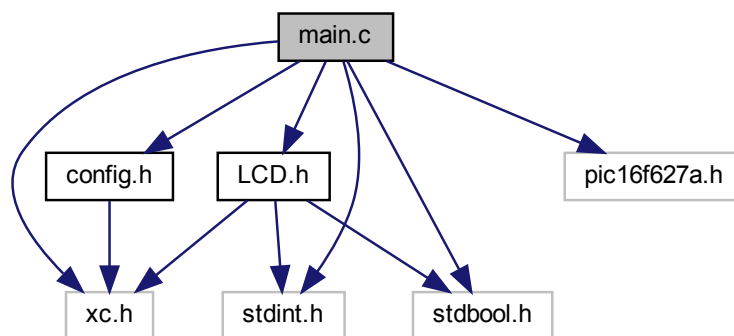
Esse é o diagrama das funções que utilizam essa função:



3.4 Referência do Arquivo main.c

```
#include "config.h"
#include <xc.h>
#include <pic16f627a.h>
#include <stdint.h>
#include <stdbool.h>
#include "LCD.h"
```

Gráfico de dependência de inclusões para main.c:



Definições e Macros

- #define `T0_cfg` 0xE0
- #define `T1_cfg` 0xA1AD

Funções

- void `ConfigureCPU` (void)
- void `main` (void)
initialize CPU and stay running in loop
- void `__interrupt` () TrataInt(void)

Variáveis

- char `LCD_Test_Char`

3.4.1 Definições e macros

3.4.1.1 T0_cfg

```
#define T0_cfg 0xE0
```

For uint8_t definition For true/false definition

Definição na linha 17 do arquivo main.c.

3.4.1.2 T1_cfg

```
#define T1_cfg 0xA1AD
```

Definição na linha 18 do arquivo main.c.

3.4.2 Funções

3.4.2.1 ConfigureCPU()

```
void ConfigureCPU (  
    void )
```

prototipos To Initialize User Ports/Peripherals/Project here

Comparators Off 111

VREF circuit powered down, no IDD drain

VREF is disconnected from RA2 pin

0=fosc/4

prescaler to T0=0

PS 000 1:2 001 1:4 010 1:8 011 1:16 100 1:32 101 1:64 110 1:128 111 1:256

desabilita temporizadores

11 = 1:8 Prescale value

0 = Internal clock (FOSC/4)

0=not external clk

1 = Enables Timer1

timer 2

T1 e T2 config

comparadores off

1 para liberar T1

enable periferal int

OPTION_REGbits.INTEDG

TRIS - define o sentido das portas set all ports as input TRISn = 0xFF; PORTA = 0x00; PORTB = 0x00;

RX=input RB1

Definição na linha 69 do arquivo main.c.

```

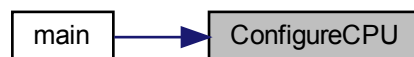
69      {
70      /** To Initialize User Ports/Peripherals/Project here */
71      CMCONbits.CM = 0b00000111; /** Comparators Off 111 */
72      // CMCONbits.CIS = 0; /** Comparator Input Switch bit */
73
74      VRCONbits.VREN = 0; /** VREF circuit powered down, no IDD drain */
75      VRCONbits.VROE = 0; /** VREF is disconnected from RA2 pin */
76
77      CCP1CONbits.CCP1M = 0b00000000;
78
79      /* 4Mhz */
80      PCONbits.OSCF = 1;
81      // timer 0
82      TMR0 = T0_cfg;
83      OPTION_REGbits.T0CS = 0; /** 0=fosc/4 */
84
85      OPTION_REGbits.PSA = 0; /** prescaler to T0=0 */
86      /**
87       * PS
88       * 000 1:2
89       * 001 1:4
90       * 010 1:8
91       * 011 1:16
92       * 100 1:32
93       * 101 1:64
94       * 110 1:128
95       * 111 1:256
96       */
97      OPTION_REGbits.PS2 = 0;
98      OPTION_REGbits.PS1 = 0;
99      OPTION_REGbits.PS0 = 1;
100
101      INTCONbits.TMR0IE = 1;
102      INTCONbits.TMR0IF = 0;
103
104      // timer 1
105      T1CON = 0x00; /** desabilita temporizadores */
106      TMR1 = T1_cfg;
107      T1CONbits.T1CKPS0 = 1; /** 11 = 1:8 Prescale value */
108      T1CONbits.T1CKPS1 = 1;
109      T1CONbits.TMR1CS = 0; /** 0 = Internal clock (FOSC/4) */
110      T1CONbits.T1OSCEN = 0; /** 0=not external clk */
111      T1CONbits.nT1SYNC = 1;
112      T1CONbits.TMR1ON = 1; /** 1 = Enables Timer1 */
113      PIR1bits.TMR1IF = 0;
114
115      /** timer 2 */
116      T2CON = 0x00;
117      T2CONbits.TMR2ON = 0;
118      PIR1bits.TMR2IF = 0;
119
120      /** T1 e T2 config */
121      PIE1bits.CCP1IE = 0;
122      PIE1bits.CMIE = 0; /** comparadores off */

```

```

123     PIELbits.EEIE = 0;
124     PIELbits.RCIE = 0;
125     PIELbits.TMR1IE = 1; /** 1 para liberar T1 */
126     PIELbits.TMR2IE = 0;
127     PIELbits.TXIE = 1;
128
129     // timer 0
130     INTCONbits.T0IE = 1;
131     INTCONbits.T0IF = 1;
132
133     INTCONbits.PEIE = 1; /** enable peripheral int */
134     INTCONbits.RBIE = 0;
135     INTCONbits.INTE = 0; /** OPTION_REGbits.INTEDG */
136     INTCONbits.INTF = 0;
137
138     /** TRIS - define o sentido das portas
139      * set all ports as input TRISn = 0XFF;
140      * PORTA = 0x00;
141      * PORTB = 0x00;
142      */
143     TRISA = 0x00;
144     TRISB = 0x00;
145     TRISBbits.TRISB1 = 1; /** RX=input RB1 */
146     OPTION_REGbits.nRBPU = 1; /**1 = PORTB pull-ups are disabled
147  }
```

Esse é o diagrama das funções que utilizam essa função:



3.4.2.2 main()

```

main (
    void )
```

initialize CPU and stay running in loop

Handles clock switching/osc initialization // 1234567890123456 LCD_Inserer_String("Ricardo de Frei-\0"); LCD_Inserer_Char(LCD_cmd_R2); LCD_Inserer_String("tas v02\0");

end of int with GIE

Definição na linha 34 do arquivo main.c.

```

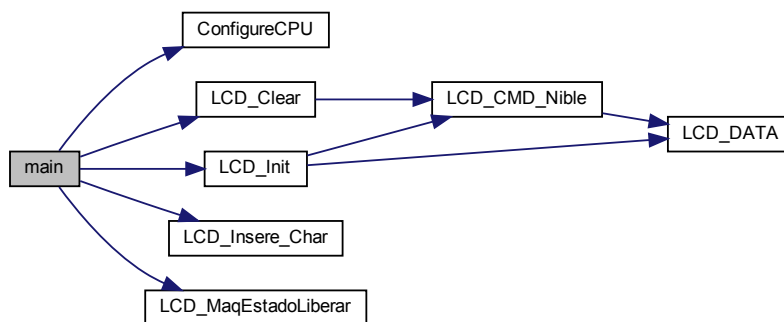
34     {
35         ConfigureCPU();
36         LCD_Init();
37         LCD_Clear();
38
39         LCD_Inserer_Char(LCD_cmd_R1);
40         LCD_Inserer_Char(LCD_cmd_2R);
41         LCD_Inserer_Char(LCD_cmd_16C);
42
43         /**
44          * // 1234567890123456
45          * LCD_Inserer_String("Ricardo de Frei-\0");
46          * LCD_Inserer_Char(LCD_cmd_R2);
47          * LCD_Inserer_String("tas v02\0");
48          */
49     }
```

```

50 //          0-----1-----2-----3-----4-----5-----6123
51 //LCD_InserString("Ricardo de Freitas 2021 08 13 entrada de dados para 4 linhas  .\0");
52
53
54 /** end of int with GIE */
55 LCD_MaqEstadoLiberar();
56 INTCONbits.GIE = 1;
57
58 while (1) {
59     NOP();
60     NOP();
61     NOP();
62 }
63 return;
64 }

```

Este é o diagrama das funções utilizadas por essa função:



3.4.2.3 __interrupt()

```
__interrupt ( )
```

executa a maquina de estado que desvazia a pilha de mensagens

overflow FF > 00

mantem o LCD trocando caracteres

jump unexpected characters

jump 00 ~ 1F

jump 80 ~ 9F

jump A6 ~ DA

jump F9 ~ xx

Definição na linha 153 do arquivo main.c.

```

153 {
154     if (INTCONbits.TMR0IE && INTCONbits.TMR0IF) {
155         /**
156          * @brief executa a maquina de estado
157          * que desvazia a pilha de mensagens

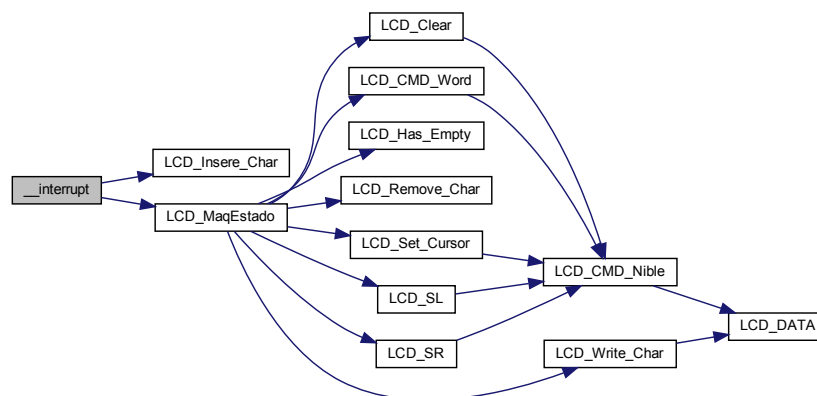
```

```

158     */
159     TMR0 = T0_cfg;
160     // LED_RAM_T0 = !LED_RAM_T0;
161     // LED_T0 = LED_RAM_T0;
162     INTCONbits.TMR0IF = 0; /** overflow FF > 00 */
163     LCD_MaqEstado();
164
165 } else if (PIR1bits.RCIF) {
166     // PIR1bits.TXIF = false;
167     PIR1bits.RCIF = 0;
168     // LED_RAM_T1 = !LED_RAM_T1;
169     // LED_T1 = LED_RAM_T1;
170
171 } else if (PIE1bits.TMR1IE && PIR1bits.TMR1IF) {
172     /**
173      * @brief mantém o LCD trocando caracteres
174      */
175     TMR1 = T1_cfg;
176     PIR1bits.TMR1IF = 0; /** overflow FF > 00
177     LCD_Test_Char++;
178     /**
179      * @brief jump unexpected characters
180      */
181     if (LCD_Test_Char < 0x20) /** jump 00 ~ 1F */
182         LCD_Test_Char = 0x20;
183
184     if (LCD_Test_Char == 0x80) /** jump 80 ~ 9F */
185         LCD_Test_Char = 0xA0;
186
187     if (LCD_Test_Char == 0xA6) /** jump A6 ~ DA */
188         LCD_Test_Char = 0xDB;
189
190     if (LCD_Test_Char == 0xF9) /** jump F9 ~ xx */
191         LCD_Test_Char = 0x20;
192
193     LCD_Inserere_Char(LCD_Test_Char);
194
195 }
196
197 }

```

Este é o diagrama das funções utilizadas por essa função:



3.4.3 Variáveis

3.4.3.1 LCD_Test_Char

```
char LCD_Test_Char
```

```
T1_cfg = 0xC1AD
```

Definição na linha 22 do arquivo main.c.

Índice Remissivo

- [_XTAL_FREQ](#)
 - [LCD.h, 23](#)
 - [__interrupt](#)
 - [main.c, 46](#)
- [config.h, 5](#)
- [ConfigureCPU](#)
 - [main.c, 43](#)
- [LCD.c, 6](#)
 - [LCD_Clear, 9](#)
 - [LCD_CMD_Nible, 8](#)
 - [LCD_CMD_Word, 9](#)
 - [LCD_Current_Row, 20](#)
 - [LCD_DATA, 7](#)
 - [LCD_Has_Empty, 16](#)
 - [LCD_Init, 11](#)
 - [LCD_Inserie_Char, 16](#)
 - [LCD_Inserie_String, 16](#)
 - [LCD_MaqEstado, 17](#)
 - [LCD_MaqEstadoLiberar, 17](#)
 - [LCD_Next_Col, 20](#)
 - [LCD_Number_Col, 20](#)
 - [LCD_Number_Row, 20](#)
 - [LCD_Remove_Char, 15](#)
 - [LCD_Set_Cursor, 10](#)
 - [LCD_SL, 13](#)
 - [LCD_SR, 14](#)
 - [LCD_Write_Char, 12](#)
 - [M_LCD, 19](#)
 - [M_LCD_inserie, 19](#)
 - [M_LCD_remove, 20](#)
 - [M_LCD_SIZE, 7](#)
 - [ME_Running, 20](#)
- [LCD.h, 21](#)
 - [_XTAL_FREQ, 23](#)
 - [LCD_Clear, 31](#)
 - [LCD_cmd_16C, 26](#)
 - [LCD_cmd_20C, 26](#)
 - [LCD_cmd_2R, 26](#)
 - [LCD_cmd_4R, 26](#)
 - [LCD_cmd_Clear, 24](#)
 - [LCD_cmd_Col00, 27](#)
 - [LCD_cmd_Col01, 27](#)
 - [LCD_cmd_Col02, 27](#)
 - [LCD_cmd_Col03, 27](#)
 - [LCD_cmd_Col04, 27](#)
 - [LCD_cmd_Col05, 28](#)
 - [LCD_cmd_Col06, 28](#)
 - [LCD_cmd_Col07, 28](#)
 - [LCD_cmd_Col08, 28](#)
 - [LCD_cmd_Col09, 28](#)
 - [LCD_cmd_Col10, 28](#)
 - [LCD_cmd_Col11, 29](#)
 - [LCD_cmd_Col12, 29](#)
 - [LCD_cmd_Col13, 29](#)
 - [LCD_cmd_Col14, 29](#)
 - [LCD_cmd_Col15, 29](#)
 - [LCD_CMD_Nible, 33](#)
 - [LCD_cmd_R1, 25](#)
 - [LCD_cmd_R2, 25](#)
 - [LCD_cmd_R3, 25](#)
 - [LCD_cmd_R4, 25](#)
 - [LCD_cmd_SL, 25](#)
 - [LCD_cmd_SR, 25](#)
 - [LCD_Cursor_blk, 27](#)
 - [LCD_Cursor_off, 26](#)
 - [LCD_Cursor_on, 26](#)
 - [LCD_D4, 24](#)
 - [LCD_D5, 24](#)
 - [LCD_D6, 24](#)
 - [LCD_D7, 24](#)
 - [LCD_DATA, 34](#)
 - [LCD_EN, 23](#)
 - [LCD_EN_Delay, 23](#)
 - [LCD_Has_Empty, 38](#)
 - [LCD_Init, 30](#)
 - [LCD_Inserie_Char, 38](#)
 - [LCD_Inserie_String, 37](#)
 - [LCD_MaqEstado, 39](#)
 - [LCD_MaqEstadoLiberar, 41](#)
 - [LCD_Remove_Char, 37](#)
 - [LCD_RS, 23](#)
 - [LCD_RW, 24](#)
 - [LCD_Set_Cursor, 35](#)
 - [LCD_SL, 32](#)
 - [LCD_SR, 32](#)
 - [LCD_Write_Char, 36](#)
- [LCD_Clear](#)
 - [LCD.c, 9](#)
 - [LCD.h, 31](#)
- [LCD_cmd_16C](#)
 - [LCD.h, 26](#)
- [LCD_cmd_20C](#)
 - [LCD.h, 26](#)
- [LCD_cmd_2R](#)
 - [LCD.h, 26](#)
- [LCD_cmd_4R](#)
 - [LCD.h, 26](#)

LCD_cmd_Clear
LCD.h, 24

LCD_cmd_Col00
LCD.h, 27

LCD_cmd_Col01
LCD.h, 27

LCD_cmd_Col02
LCD.h, 27

LCD_cmd_Col03
LCD.h, 27

LCD_cmd_Col04
LCD.h, 27

LCD_cmd_Col05
LCD.h, 28

LCD_cmd_Col06
LCD.h, 28

LCD_cmd_Col07
LCD.h, 28

LCD_cmd_Col08
LCD.h, 28

LCD_cmd_Col09
LCD.h, 28

LCD_cmd_Col10
LCD.h, 28

LCD_cmd_Col11
LCD.h, 29

LCD_cmd_Col12
LCD.h, 29

LCD_cmd_Col13
LCD.h, 29

LCD_cmd_Col14
LCD.h, 29

LCD_cmd_Col15
LCD.h, 29

LCD_CMD_Nible
LCD.c, 8
LCD.h, 33

LCD_cmd_R1
LCD.h, 25

LCD_cmd_R2
LCD.h, 25

LCD_cmd_R3
LCD.h, 25

LCD_cmd_R4
LCD.h, 25

LCD_cmd_SL
LCD.h, 25

LCD_cmd_SR
LCD.h, 25

LCD_CMD_Word
LCD.c, 9

LCD_Current_Row
LCD.c, 20

LCD_Cursor_blk
LCD.h, 27

LCD_Cursor_off
LCD.h, 26

LCD_Cursor_on
LCD.h, 26

LCD_D4
LCD.h, 24

LCD_D5
LCD.h, 24

LCD_D6
LCD.h, 24

LCD_D7
LCD.h, 24

LCD_DATA
LCD.c, 7
LCD.h, 34

LCD_EN
LCD.h, 23

LCD_EN_Delay
LCD.h, 23

LCD_Has_Empty
LCD.c, 16
LCD.h, 38

LCD_Init
LCD.c, 11
LCD.h, 30

LCD_Inserere_Char
LCD.c, 16
LCD.h, 38

LCD_Inserere_String
LCD.c, 16
LCD.h, 37

LCD_MaqEstado
LCD.c, 17
LCD.h, 39

LCD_MaqEstadoLiberar
LCD.c, 17
LCD.h, 41

LCD_Next_Col
LCD.c, 20

LCD_Number_Col
LCD.c, 20

LCD_Number_Row
LCD.c, 20

LCD_Remove_Char
LCD.c, 15
LCD.h, 37

LCD_RS
LCD.h, 23

LCD_RW
LCD.h, 24

LCD_Set_Cursor
LCD.c, 10
LCD.h, 35

LCD_SL
LCD.c, 13
LCD.h, 32

LCD_SR
LCD.c, 14
LCD.h, 32

LCD_Test_Char
main.c, 47

LCD_Write_Char
 LCD.c, [12](#)
 LCD.h, [36](#)

M_LCD
 LCD.c, [19](#)

M_LCD_insere
 LCD.c, [19](#)

M_LCD_remove
 LCD.c, [20](#)

M_LCD_SIZE
 LCD.c, [7](#)

main
 main.c, [45](#)

main.c, [42](#)
 __interrupt, [46](#)
 ConfigureCPU, [43](#)
 LCD_Test_Char, [47](#)
 main, [45](#)
 T0_cfg, [43](#)
 T1_cfg, [43](#)

ME_Running
 LCD.c, [20](#)

T0_cfg
 main.c, [43](#)

T1_cfg
 main.c, [43](#)