# LCD library (Source: )

1. ***//i2c-2004.h***
2. **#include "stm32f4xx\_hal.h"**
4. **void lcd\_init (void); *// initialize lcd***
5. **void lcd\_send\_cmd (char cmd); *// send command to the lcd***
6. **void lcd\_send\_data (char data); *// send data to the lcd***
7. **void lcd\_send\_string (char \*str); *// send string to the lcd***
8. **void lcd\_clear (void);**
10. ***//i2c-2004.c***
11. ***/\*\* Put this in the src folder \*\*/***
12. **#include "i2c-2004.h"**
13. **extern I2C\_HandleTypeDef hi2c1; *// change your handler here accordingly***
14. **#define SLAVE\_ADDRESS\_LCD 0x7E//0x4E // change this according to ur setup**
16. **void lcd\_send\_cmd (char cmd)**
17. **{**
18. **char data\_u, data\_l;**
19. **uint8\_t data\_t[4];**
20. **data\_u = (cmd&0xf0);**
21. **data\_l = ((cmd<<4)&0xf0);**
22. **data\_t[0] = data\_u|0x0C; *//en=1, rs=0***
23. **data\_t[1] = data\_u|0x08; *//en=0, rs=0***
24. **data\_t[2] = data\_l|0x0C; *//en=1, rs=0***
25. **data\_t[3] = data\_l|0x08; *//en=0, rs=0***
26. **HAL\_I2C\_Master\_Transmit (&hi2c1, SLAVE\_ADDRESS\_LCD,(uint8\_t \*) data\_t, 4, 100);**
27. **}**
29. **void lcd\_send\_data (char data)**
30. **{**
31. **char data\_u, data\_l;**
32. **uint8\_t data\_t[4];**
33. **data\_u = (data&0xf0);**
34. **data\_l = ((data<<4)&0xf0);**
35. **data\_t[0] = data\_u|0x0D; *//en=1, rs=1***
36. **data\_t[1] = data\_u|0x09; *//en=0, rs=1***
37. **data\_t[2] = data\_l|0x0D; *//en=1, rs=1***
38. **data\_t[3] = data\_l|0x09; *//en=0, rs=1***
39. **HAL\_I2C\_Master\_Transmit (&hi2c1, SLAVE\_ADDRESS\_LCD,(uint8\_t \*) data\_t, 4, 100);**
40. **}**
42. **void lcd\_clear (void)**
43. **{**
44. **lcd\_send\_cmd (0x00);**
45. **for (int i=0; i<100; i++)**
46. **{**
47. **lcd\_send\_data (' ');**
48. **}**
49. **}**
51. **void lcd\_init (void)**
52. **{**
53. ***// 4 bit initialisation***
54. **HAL\_Delay(50); *// wait for >40ms***
55. **lcd\_send\_cmd (0x30);**
56. **HAL\_Delay(5); *// wait for >4.1ms***
57. **lcd\_send\_cmd (0x30);**
58. **HAL\_Delay(1); *// wait for >100us***
59. **lcd\_send\_cmd (0x30);**
60. **HAL\_Delay(10);**
61. **lcd\_send\_cmd (0x20); *// 4bit mode***
62. **HAL\_Delay(10);**
64. ***// dislay initialisation***
65. **lcd\_send\_cmd (0x28); *// Function set --> DL=0 (4 bit mode), N = 1 (2 line display) F = 0 (5x8 characters)***
66. **HAL\_Delay(1);**
67. **lcd\_send\_cmd (0x08); *//Display on/off control --> D=0,C=0, B=0 ---> display off***
68. **HAL\_Delay(1);**
69. **lcd\_send\_cmd (0x01); *// clear display***
70. **HAL\_Delay(1);**
71. **HAL\_Delay(1);**
72. **lcd\_send\_cmd (0x06); *//Entry mode set --> I/D = 1 (increment cursor) & S = 0 (no shift)***
73. **HAL\_Delay(1);**
74. **lcd\_send\_cmd (0x0C); *//Display on/off control --> D = 1, C and B = 0. (Cursor and blink, last two bits)***
75. **}**
77. **void lcd\_send\_string (char \*str)**
78. **{**
79. **while (\*str) lcd\_send\_data (\*str++);**
80. **}**

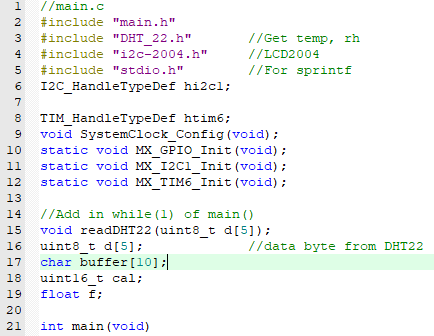
# DHT22 library (Source: )

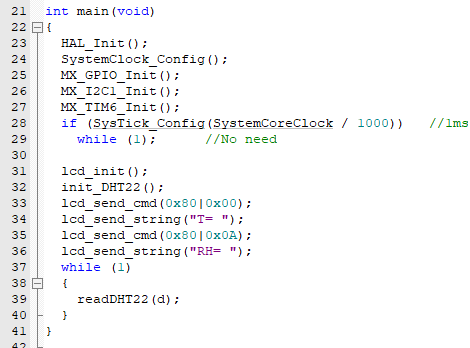
1. ***//DHT\_22.h***
2. **#include "stm32f4xx\_hal.h"**
3. ***//Pin read/Write Data from DHT11 Sensor***
4. **#define DHT11\_PORT GPIOA**
5. **#define DHT11\_PIN GPIO\_PIN\_1**
7. ***//Add in main() (Note: before while(1))***
8. **void init\_DHT22(void);**
9. **void delayus(uint16\_t time); *//Delay in microsec.***
10. **void Set\_Pin\_Output (GPIO\_TypeDef \*GPIOx, uint16\_t GPIO\_Pin);**
11. **void Set\_Pin\_Input (GPIO\_TypeDef \*GPIOx, uint16\_t GPIO\_Pin);**
12. ***//Read 1 byte (in 5 bytes transmission)***
13. **uint8\_t DHT22\_Read (void);**
14. ***//Check DHT connection and data valid???***
15. **uint8\_t DHT22\_Check\_Response(uint8\_t d[5]);**
17. ***//DHT\_22.c***
18. **#include "DHT\_22.h"**
19. **extern TIM\_HandleTypeDef htim6;**
21. **void init\_DHT22(void)**
22. **{**
23. **HAL\_TIM\_Base\_Start(&htim6);**
24. **}**
26. **void delayus(uint16\_t time) *//Code 1: Timer 6***
27. **{**
28. ***//change your code here for the delay in microseconds***
29. **\_\_HAL\_TIM\_SET\_COUNTER(&htim6, 0);**
30. **while ((\_\_HAL\_TIM\_GET\_COUNTER(&htim6))<time);**
31. **}**
33. **void Set\_Pin\_Output (GPIO\_TypeDef \*GPIOx, uint16\_t GPIO\_Pin)**
34. **{**
35. **GPIO\_InitTypeDef GPIO\_InitStruct = {0};**
36. **GPIO\_InitStruct.Pin = GPIO\_Pin;**
37. **GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;**
38. **GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_VERY\_HIGH;**
39. **HAL\_GPIO\_Init(GPIOx, &GPIO\_InitStruct);**
40. **}**
42. **void Set\_Pin\_Input (GPIO\_TypeDef \*GPIOx, uint16\_t GPIO\_Pin)**
43. **{**
44. **GPIO\_InitTypeDef GPIO\_InitStruct = {0};**
45. **GPIO\_InitStruct.Pin = GPIO\_Pin;**
46. **GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;**
47. **GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_VERY\_HIGH;**
48. **HAL\_GPIO\_Init(GPIOx, &GPIO\_InitStruct);**
49. **}**
51. **uint8\_t DHT22\_Read (void)**
52. **{**
53. **uint8\_t i,j;**
54. **for (j=0;j<8;j++)**
55. **{**
56. **while (!(HAL\_GPIO\_ReadPin (DHT11\_PORT, DHT11\_PIN)));**
57. **delayus(30); *//Same as DHT11 and DHT22***
58. **if ((HAL\_GPIO\_ReadPin (DHT11\_PORT, DHT11\_PIN)))**
59. **{**
60. **while (HAL\_GPIO\_ReadPin (DHT11\_PORT, DHT11\_PIN));**
61. **i |= (1<<(7-j));**
62. **}**
63. **else i &= ~(1<<(7-j));**
64. **}**
65. **return i;**
66. **}**
68. **uint8\_t DHT22\_Check\_Response(uint8\_t d[])**
69. **{**
70. **HAL\_GPIO\_WritePin (DHT11\_PORT, DHT11\_PIN, GPIO\_PIN\_RESET);**
71. **delayus(1000); *//For DHT22, DHT11 takes 20000us***
72. **HAL\_GPIO\_WritePin (DHT11\_PORT, DHT11\_PIN, GPIO\_PIN\_SET);**
73. **Set\_Pin\_Input(DHT11\_PORT, DHT11\_PIN);**
74. **delayus(40);**
75. **if (!(HAL\_GPIO\_ReadPin (DHT11\_PORT, DHT11\_PIN)))**
76. **{**
77. **while (!(HAL\_GPIO\_ReadPin (DHT11\_PORT, DHT11\_PIN)));**
78. **while ((HAL\_GPIO\_ReadPin (DHT11\_PORT, DHT11\_PIN)));**
79. **d[0] = DHT22\_Read();**
80. **d[1] = DHT22\_Read();**
81. **d[2] = DHT22\_Read();**
82. **d[3] = DHT22\_Read();**
83. **d[4] = DHT22\_Read();**
84. **Set\_Pin\_Output(DHT11\_PORT, DHT11\_PIN);**
85. **HAL\_GPIO\_WritePin (DHT11\_PORT, DHT11\_PIN, GPIO\_PIN\_SET);**
86. **if(d[4] == d[3]+d[2]+d[1]+d[0])**
87. **return 1;**
88. **else return 0;**
89. **}**
90. **else**
91. **{**
92. **Set\_Pin\_Output(DHT11\_PORT, DHT11\_PIN);**
93. **HAL\_GPIO\_WritePin (DHT11\_PORT, DHT11\_PIN, GPIO\_PIN\_SET);**
94. **return 0;**
95. **}**
96. **}**

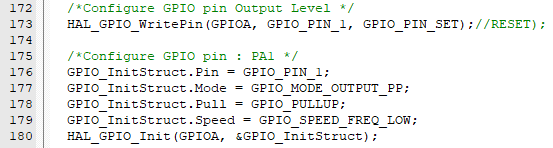
# Main.c:

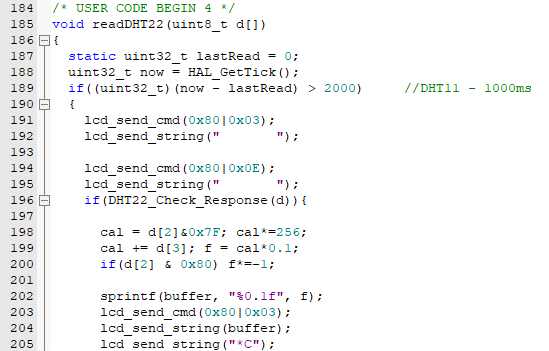
1. ***//main.c***
2. **#include "main.h"**
3. **#include "DHT\_22.h" //Get temp, rh**
4. **#include "i2c-2004.h" //LCD2004**
5. **#include "stdio.h" //For sprintf**
6. **I2C\_HandleTypeDef hi2c1;**
8. **TIM\_HandleTypeDef htim6;**
9. **void SystemClock\_Config(void);**
10. **static void MX\_GPIO\_Init(void);**
11. **static void MX\_I2C1\_Init(void);**
12. **static void MX\_TIM6\_Init(void);**
14. ***//Add in while(1) of main()***
15. **void readDHT22(uint8\_t d[5]);**
16. **uint8\_t d[5]; *//data byte from DHT22***
17. **char buffer[10];**
18. **uint16\_t cal;**
19. **float f;**
21. **int main(void)**
22. **{**
23. **HAL\_Init();**
24. **SystemClock\_Config();**
25. **MX\_GPIO\_Init();**
26. **MX\_I2C1\_Init();**
27. **MX\_TIM6\_Init();**
28. **if (SysTick\_Config(SystemCoreClock / 1000)) *//1ms***
29. **while (1); *//No need***
31. **lcd\_init();**
32. **init\_DHT22();**
33. **lcd\_send\_cmd(0x80|0x00);**
34. **lcd\_send\_string("T= ");**
35. **lcd\_send\_cmd(0x80|0x0A);**
36. **lcd\_send\_string("RH= ");**
37. **while (1)**
38. **{**
39. **readDHT22(d);**
40. **}**
41. **}**
43. ***/\*\****
44. ***\* @brief System Clock Configuration***
45. ***\* @retval None***
46. ***\*/***
47. **void SystemClock\_Config(void)**
48. **{**
49. **RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};**
50. **RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};**
52. ***/\*\* Configure the main internal regulator output voltage***
53. ***\*/***
54. **\_\_HAL\_RCC\_PWR\_CLK\_ENABLE();**
55. **\_\_HAL\_PWR\_VOLTAGESCALING\_CONFIG(PWR\_REGULATOR\_VOLTAGE\_SCALE1);**
56. ***/\*\* Initializes the RCC Oscillators according to the specified parameters***
57. ***\* in the RCC\_OscInitTypeDef structure.***
58. ***\*/***
59. **RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSE;**
60. **RCC\_OscInitStruct.HSEState = RCC\_HSE\_ON;**
61. **RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_ON;**
62. **RCC\_OscInitStruct.PLL.PLLSource = RCC\_PLLSOURCE\_HSE;**
63. **RCC\_OscInitStruct.PLL.PLLM = 8;**
64. **RCC\_OscInitStruct.PLL.PLLN = 336;**
65. **RCC\_OscInitStruct.PLL.PLLP = RCC\_PLLP\_DIV2;**
66. **RCC\_OscInitStruct.PLL.PLLQ = 4;**
67. **if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)**
68. **{**
69. **Error\_Handler();**
70. **}**
71. ***/\*\* Initializes the CPU, AHB and APB buses clocks***
72. ***\*/***
73. **RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK**
74. **|RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;**
75. **RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_PLLCLK;**
76. **RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV1;**
77. **RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV4;**
78. **RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV2;**
80. **if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_5) != HAL\_OK)**
81. **{**
82. **Error\_Handler();**
83. **}**
84. **}**
86. ***/\*\****
87. ***\* @brief I2C1 Initialization Function***
88. ***\* @param None***
89. ***\* @retval None***
90. ***\*/***
91. **static void MX\_I2C1\_Init(void)**
92. **{**
94. ***/\* USER CODE BEGIN I2C1\_Init 0 \*/***
96. ***/\* USER CODE END I2C1\_Init 0 \*/***
98. ***/\* USER CODE BEGIN I2C1\_Init 1 \*/***
100. ***/\* USER CODE END I2C1\_Init 1 \*/***
101. **hi2c1.Instance = I2C1;**
102. **hi2c1.Init.ClockSpeed = 100000;**
103. **hi2c1.Init.DutyCycle = I2C\_DUTYCYCLE\_2;**
104. **hi2c1.Init.OwnAddress1 = 0;**
105. **hi2c1.Init.AddressingMode = I2C\_ADDRESSINGMODE\_7BIT;**
106. **hi2c1.Init.DualAddressMode = I2C\_DUALADDRESS\_DISABLE;**
107. **hi2c1.Init.OwnAddress2 = 0;**
108. **hi2c1.Init.GeneralCallMode = I2C\_GENERALCALL\_DISABLE;**
109. **hi2c1.Init.NoStretchMode = I2C\_NOSTRETCH\_DISABLE;**
110. **if (HAL\_I2C\_Init(&hi2c1) != HAL\_OK)**
111. **{**
112. **Error\_Handler();**
113. **}**
114. ***/\* USER CODE BEGIN I2C1\_Init 2 \*/***
116. ***/\* USER CODE END I2C1\_Init 2 \*/***
118. **}**
120. ***/\*\****
121. ***\* @brief TIM6 Initialization Function***
122. ***\* @param None***
123. ***\* @retval None***
124. ***\*/***
125. **static void MX\_TIM6\_Init(void)**
126. **{**
128. ***/\* USER CODE BEGIN TIM6\_Init 0 \*/***
130. ***/\* USER CODE END TIM6\_Init 0 \*/***
132. **TIM\_MasterConfigTypeDef sMasterConfig = {0};**
134. ***/\* USER CODE BEGIN TIM6\_Init 1 \*/***
136. ***/\* USER CODE END TIM6\_Init 1 \*/***
137. **htim6.Instance = TIM6;**
138. **htim6.Init.Prescaler = 168-1;**
139. **htim6.Init.CounterMode = TIM\_COUNTERMODE\_UP;**
140. **htim6.Init.Period = 0xFFFF;**
141. **htim6.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_DISABLE;**
142. **if (HAL\_TIM\_Base\_Init(&htim6) != HAL\_OK)**
143. **{**
144. **Error\_Handler();**
145. **}**
146. **sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;**
147. **sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;**
148. **if (HAL\_TIMEx\_MasterConfigSynchronization(&htim6, &sMasterConfig) != HAL\_OK)**
149. **{**
150. **Error\_Handler();**
151. **}**
152. ***/\* USER CODE BEGIN TIM6\_Init 2 \*/***
154. ***/\* USER CODE END TIM6\_Init 2 \*/***
156. **}**
158. ***/\*\****
159. ***\* @brief GPIO Initialization Function***
160. ***\* @param None***
161. ***\* @retval None***
162. ***\*/***
163. **static void MX\_GPIO\_Init(void)**
164. **{**
165. **GPIO\_InitTypeDef GPIO\_InitStruct = {0};**
167. ***/\* GPIO Ports Clock Enable \*/***
168. **\_\_HAL\_RCC\_GPIOH\_CLK\_ENABLE();**
169. **\_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();**
170. **\_\_HAL\_RCC\_GPIOB\_CLK\_ENABLE();**
172. ***/\*Configure GPIO pin Output Level \*/***
173. **HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1, GPIO\_PIN\_SET);*//RESET);***
175. ***/\*Configure GPIO pin : PA1 \*/***
176. **GPIO\_InitStruct.Pin = GPIO\_PIN\_1;**
177. **GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;**
178. **GPIO\_InitStruct.Pull = GPIO\_PULLUP;**
179. **GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;**
180. **HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);**
182. **}**
184. ***/\* USER CODE BEGIN 4 \*/***
185. **void readDHT22(uint8\_t d[])**
186. **{**
187. **static uint32\_t lastRead = 0;**
188. **uint32\_t now = HAL\_GetTick();**
189. **if((uint32\_t)(now - lastRead) > 2000) *//DHT11 - 1000ms***
190. **{**
191. **lcd\_send\_cmd(0x80|0x03);**
192. **lcd\_send\_string(" ");**
194. **lcd\_send\_cmd(0x80|0x0E);**
195. **lcd\_send\_string(" ");**
196. **if(DHT22\_Check\_Response(d)){**
198. **cal = d[2]&0x7F; cal\*=256;**
199. **cal += d[3]; f = cal\*0.1;**
200. **if(d[2] & 0x80) f\*=-1;**
202. **sprintf(buffer, "%0.1f", f);**
203. **lcd\_send\_cmd(0x80|0x03);**
204. **lcd\_send\_string(buffer);**
205. **lcd\_send\_string("\*C");**
207. **cal = d[0]; cal\*=256;**
208. **cal += d[1]; f = cal\*0.1;**
210. **sprintf(buffer, "%0.1f", f);**
211. **lcd\_send\_cmd(0x80|0x0E);**
212. **lcd\_send\_string(buffer);**
213. **lcd\_send\_string("%");**
214. **}**
215. **else {**
216. **lcd\_send\_cmd(0x80|0x03);**
217. **lcd\_send\_string("ERROR");**
218. **lcd\_send\_cmd(0x80|0x0E);**
219. **lcd\_send\_string("ERROR");**
220. **}**
221. **lastRead = HAL\_GetTick();**
222. **}**
223. **}**
225. ***/\* USER CODE END 4 \*/***
227. ***/\*\****
228. ***\* @brief This function is executed in case of error occurrence.***
229. ***\* @retval None***
230. ***\*/***
231. **void Error\_Handler(void)**
232. **{**
233. ***/\* USER CODE BEGIN Error\_Handler\_Debug \*/***
234. ***/\* User can add his own implementation to report the HAL error return state \*/***
235. **while(1)**
236. **{**
237. **}**
238. ***/\* USER CODE END Error\_Handler\_Debug \*/***
239. **}**
241. **#ifdef USE\_FULL\_ASSERT**
242. ***/\*\****
243. ***\* @brief Reports the name of the source file and the source line number***
244. ***\* where the assert\_param error has occurred.***
245. ***\* @param file: pointer to the source file name***
246. ***\* @param line: assert\_param error line source number***
247. ***\* @retval None***
248. ***\*/***
249. **void assert\_failed(uint8\_t \*file, uint32\_t line)**
250. **{**
251. ***/\* USER CODE BEGIN 6 \*/***
252. ***/\* User can add his own implementation to report the file name and line number,***
253. ***tex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) \*/***
254. ***/\* USER CODE END 6 \*/***
255. **}**
256. **#endif /\* USE\_FULL\_ASSERT \*/**
258. ***/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT STMicroelectronics \*\*\*\*\*END OF FILE\*\*\*\*/***

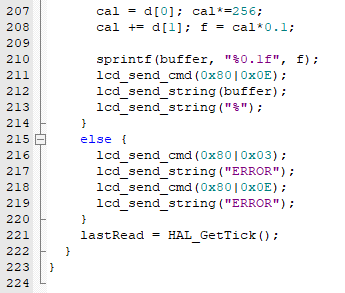
Comments:











System clk là 168Mhz -> APB1,2 cũng là 168Mhz (xem trong Cube)

Timer\_6: Prescaler – (168-1), Mode – Up, Period – (0xFFFF-1)