

L.A.M.P - PSoC Master

Genereret af Doxygen 1.8.11

Indhold

1	Indeks over datastrukturer	2
1.1	Datastrukturer	2
2	Fil-indeks	2
2.1	Filoversigt	2
3	Datastruktur-documentation	3
3.1	Data Klasse-reference	3
3.1.1	Detaljeret beskrivelse	4
3.1.2	Dokumentation af medlemsfunktioner	4
3.2	Handler Klasse-reference	5
3.2.1	Detaljeret beskrivelse	5
3.2.2	Dokumentation af medlemsfunktioner	5
3.3	I2C Klasse-reference	8
3.3.1	Detaljeret beskrivelse	9
3.3.2	Dokumentation af medlemsfunktioner	9
3.4	LCD Klasse-reference	15
3.4.1	Detaljeret beskrivelse	16
3.4.2	Dokumentation af medlemsfunktioner	16
3.4.3	Felt-dokumentation	17
3.5	LED Klasse-reference	17
3.5.1	Detaljeret beskrivelse	18
3.5.2	Dokumentation af medlemsfunktioner	18
3.6	Queue Klasse-reference	19
3.6.1	Detaljeret beskrivelse	20
3.6.2	Dokumentation af medlemsfunktioner	20
3.6.3	Felt-dokumentation	26
3.7	SPI Klasse-reference	27
3.7.1	Detaljeret beskrivelse	28
3.7.2	Dokumentation af medlemsfunktioner	28

4	Fil-dokumentation	30
4.1	cyapicalbacks.h filreference	30
4.2	data.c filreference	30
4.2.1	Detaljeret beskrivelse	31
4.3	data.h filreference	31
4.3.1	Detaljeret beskrivelse	32
4.3.2	Datastruktur-dokumentation	32
4.3.3	Funktions-dokumentation	33
4.3.4	Variabel-dokumentation	33
4.4	handler.c filreference	34
4.4.1	Detaljeret beskrivelse	34
4.5	handler.h filreference	34
4.5.1	Detaljeret beskrivelse	36
4.5.2	#Define-dokumentation	36
4.5.3	Funktions-dokumentation	39
4.6	i2c.c filreference	39
4.6.1	Detaljeret beskrivelse	40
4.7	i2c.h filreference	40
4.7.1	Detaljeret beskrivelse	41
4.7.2	#Define-dokumentation	41
4.7.3	Funktions-dokumentation	43
4.8	lcd.c filreference	43
4.8.1	Detaljeret beskrivelse	43
4.9	lcd.h filreference	44
4.9.1	Detaljeret beskrivelse	44
4.9.2	Funktions-dokumentation	45
4.10	led.c filreference	45
4.10.1	Detaljeret beskrivelse	45
4.11	led.h filreference	45
4.11.1	Detaljeret beskrivelse	46

4.11.2	#Define-dokumentation	47
4.11.3	Funktions-dokumentation	47
4.12	main.c filreference	47
4.12.1	Detaljeret beskrivelse	48
4.12.2	Funktions-dokumentation	48
4.13	Nokia5110LCD.c filreference	49
4.13.1	Detaljeret beskrivelse	50
4.13.2	Funktions-dokumentation	50
4.13.3	Variabel-dokumentation	56
4.14	Nokia5110LCD.h filreference	57
4.14.1	Detaljeret beskrivelse	59
4.14.2	#Define-dokumentation	59
4.14.3	Funktions-dokumentation	66
4.15	queue.c filreference	72
4.15.1	Detaljeret beskrivelse	73
4.15.2	Datastruktur-dokumentation	73
4.15.3	Funktions-dokumentation	74
4.16	queue.h filreference	75
4.16.1	Detaljeret beskrivelse	76
4.16.2	Datastruktur-dokumentation	76
4.16.3	Funktions-dokumentation	77
4.17	spi.c filreference	77
4.17.1	Detaljeret beskrivelse	78
4.18	spi.h filreference	78
4.18.1	Detaljeret beskrivelse	79
4.18.2	#Define-dokumentation	79
4.18.3	Funktions-dokumentation	79

1 Indeks over datastrukturer

1.1 Datastrukturer

Her er datastrukturerne med korte beskrivelser:

Data	
Data class	??
Handler	
Handler class	??
I2C	
I2C class	??
LCD	
LCD class	??
LED	
LED class	??
Queue	
Queue class	??
SPI	
SPI class	??

2 Fil-indeks

2.1 Filoversigt

Her er en liste over alle filer med korte beskrivelser:

cyapicallbacks.h	??
data.c	
Data modul	??
data.h	
Data modul	??
handler.c	
Handler modul	??
handler.h	
Handler modul	??
i2c.c	
I2C modul	??
i2c.h	
I2C modul	??
lcd.c	
LCD modul	??

lcd.h		
LCD modul		??
led.c		
LED modul		??
led.h		
LED modul		??
main.c		
Hovedprogram		??
Nokia5110LCD.c		
Nokia5110LCD Modul (Impoteret)		??
Nokia5110LCD.h		
Nokia5110LCD Modul (Impoteret)		??
queue.c		
Queue modul		??
queue.h		
Queue modul		??
spi.c		
SPI modul		??
spi.h		
SPI modul		??

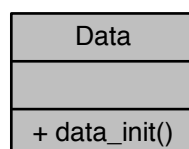
3 Datastruktur-documentation

3.1 Data Klasse-reference

[Data](#) class.

```
#include <data.h>
```

Samarbejdsdiagram for Data:



Offentlige metoder

- void [data_init](#) ()
Initialiser data modulet.

3.1.1 Detaljeret beskrivelse

[Data](#) class.

Indeholder data hentet fra PSoC-XY, -Z og -Sensor.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

3.1.2 Dokumentation af medlemsfunktioner

3.1.2.1 void data_init (void)

Initialiser data modulet.

Initialiser data structen med 0 værdier.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 20 i filen data.c.

Indeholder referencer til DataMaster::bVal, dataMaster, DataMaster::gVal, DataMaster::rVal, DataMaster::xVal, DataMaster::yVal og DataMaster::zVal.

Refereret til af main().

```
21 {  
22     dataMaster.xVal = 0;  
23     dataMaster.yVal = 0;  
24     dataMaster.zVal = 0;  
25     dataMaster.rVal = 0;  
26     dataMaster.gVal = 0;  
27     dataMaster.bVal = 0;  
28 }
```

Her er kalder-grafen for denne funktion:



Dokumentationen for denne klasse blev genereret ud fra filerne:

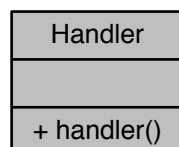
- [data.h](#)
- [data.c](#)

3.2 Handler Klasse-reference

[Handler](#) class.

```
#include <handler.h>
```

Samarbejdsdiagram for Handler:



Offentlige metoder

- void [handler](#) (uint8 cmd, uint8 val)
Håndter kommando med tilhørende værdi.

3.2.1 Detaljeret beskrivelse

[Handler](#) class.

Håndtere indkommende kommandoer med tilhørende værdier.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

3.2.2 Dokumentation af medlemsfunktioner

3.2.2.1 void handler (uint8 cmd, uint8 val)

Håndter kommando med tilhørende værdi.

Fortager en defineret handling ud fra den modtaget kommando med den tilhørende værdi.

Parametre

in	<i>cmd</i>	Er den modtaget kommando.
in	<i>val</i>	Er den tilhørende værdi.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 26 i filen handler.c.

Indeholder referencer til DataMaster::bVal, CMD_DISTANCE_ALRT, CMD_GET_BLUE_VAL, CMD_GET_DISTANCE_STS, CMD_GET_GREEN_VAL, CMD_GET_LUMEN_VAL, CMD_GET_MOVEMENT_STS, CMD_GET_POWER_STS, CMD_GET_RED_VAL, CMD_GET_X_POS, CMD_GET_Y_POS, CMD_GET_Z_POS, CMD_MOVEMENT_ALRT, CMD_SET_BLUE_VAL, CMD_SET_DISTANCE_STS, CMD_SET_GREEN_VAL, CMD_SET_LUMEN_VAL, CMD_SET_MOVEMENT_STS, CMD_SET_POWER_STS, CMD_SET_RED_VAL, CMD_SET_X_POS, CMD_SET_Y_POS, CMD_SET_Z_POS, CMD_X_CAL, CMD_X_STP, CMD_Y_CAL, CMD_Y_STP, CMD_Z_CAL, CMD_Z_STP, dataMaster, DataMaster::gVal, I2C::i2c_getPacket(), I2C::i2c_setPacket(), PSoC_Sensor, PSoC_XY, PSoC_Z, DataMaster::rVal, DataMaster::xVal, DataMaster::yVal og DataMaster::zVal.

Refereret til af main().

```

27 {
28     DEBUG_PutString("H=: cmd: ");
29     DEBUG_PutHexByte(cmd);
30     DEBUG_PutString(" val: ");
31     DEBUG_PutHexByte(val);
32     DEBUG_PutCRLF();
33
34     switch (cmd)
35     {
36     case 0x01 :
37         i2c_getPacket(PSoC_XY, CMD_GET_X_POS, &
dataMaster.xVal);
38         i2c_getPacket(PSoC_XY, CMD_GET_Y_POS, &
dataMaster.yVal);
39         i2c_getPacket(PSoC_Z, CMD_GET_Z_POS, &
dataMaster.zVal);
40         break;
41     case 0x03 :
42         i2c_getPacket(PSoC_Sensor, CMD_GET_RED_VAL, &
dataMaster.rVal);
43         i2c_getPacket(PSoC_Sensor, CMD_GET_BLUE_VAL, &
dataMaster.gVal);
44         i2c_getPacket(PSoC_Sensor, CMD_GET_GREEN_VAL, &
dataMaster.bVal);
45         break;
46     case CMD_SET_X_POS :
47         i2c_setPacket(PSoC_XY, cmd, val);
48         break;
49     case CMD_SET_Y_POS :
50         i2c_setPacket(PSoC_XY, cmd, val);
51         break;
52     case CMD_GET_X_POS :
53         /* Håndteres i SPI modulet */
54         break;
55     case CMD_GET_Y_POS :
56         /* Håndteres i SPI modulet */
57         break;
58     case CMD_X_STP :
59         i2c_setPacket(PSoC_XY, cmd, val);
60         break;
61     case CMD_Y_STP :
62         i2c_setPacket(PSoC_XY, cmd, val);
63         break;
64     case CMD_X_CAL :
65         i2c_setPacket(PSoC_XY, cmd, val);
66         break;
67     case CMD_Y_CAL :
68         i2c_setPacket(PSoC_XY, cmd, val);
69         break;
70     case CMD_SET_Z_POS :
71         i2c_setPacket(PSoC_Z, cmd, val);
72         break;
73     case CMD_GET_Z_POS :
74         /* Håndteres i SPI modulet */
75         break;
76     case CMD_Z_STP :
77         i2c_setPacket(PSoC_Z, cmd, val);
78         break;
79     case CMD_Z_CAL :
80         i2c_setPacket(PSoC_Z, cmd, val);
81         break;

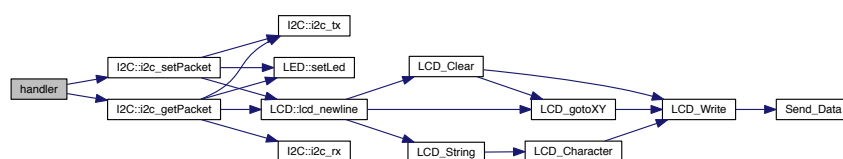
```

```

82     case CMD_SET_RED_VAL :
83         i2c_setPacket(PSoC_Sensor, cmd, val);
84         break;
85     case CMD_SET_GREEN_VAL :
86         i2c_setPacket(PSoC_Sensor, cmd, val);
87         break;
88     case CMD_SET_BLUE_VAL :
89         i2c_setPacket(PSoC_Sensor, cmd, val);
90         break;
91     case CMD_SET_LUMEN_VAL :
92         i2c_setPacket(PSoC_Sensor, cmd, val);
93         break;
94     case CMD_SET_POWER_STS :
95         i2c_setPacket(PSoC_Sensor, cmd, val);
96         break;
97     case CMD_GET_RED_VAL :
98         /* Håndteres i SPI modulet */
99         break;
100    case CMD_GET_GREEN_VAL :
101        /* Håndteres i SPI modulet */
102        break;
103    case CMD_GET_BLUE_VAL :
104        /* Håndteres i SPI modulet */
105        break;
106    case CMD_GET_LUMEN_VAL :
107        /* Håndteres i SPI modulet */
108        break;
109    case CMD_GET_POWER_STS :
110        /* Håndteres i SPI modulet */
111        break;
112    case CMD_SET_DISTANCE_STS :
113        i2c_setPacket(PSoC_Sensor, cmd, val);
114        break;
115    case CMD_SET_MOVEMENT_STS :
116        i2c_setPacket(PSoC_Sensor, cmd, val);
117        break;
118    case CMD_GET_DISTANCE_STS :
119        /* Håndteres i SPI modulet */
120        break;
121    case CMD_GET_MOVEMENT_STS :
122        /* Håndteres i SPI modulet */
123        break;
124    case CMD_DISTANCE_ALRT :
125        handler(CMD_X_STP, val);
126        handler(CMD_Y_STP, val);
127        handler(CMD_Z_STP, val);
128        break;
129    case CMD_MOVEMENT_ALRT :
130        handler(CMD_SET_POWER_STS, val);
131        break;
132    default :
133        break;
134 }
135 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



Dokumentationen for denne klasse blev genereret ud fra filerne:

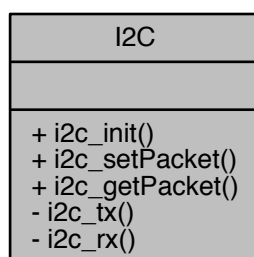
- [handler.h](#)
- [handler.c](#)

3.3 I2C Klasse-reference

I2C class.

```
#include <i2c.h>
```

Samarbejdsdiagram for I2C:



Offentlige metoder

- void [i2c_init](#) ()
Initialiser I2C modulet.
- void [i2c_setPacket](#) (uint8 i2cAddr, uint8 i2cCmd, uint8 i2cVal)
Sender en I2C pakke.
- void [i2c_getPacket](#) (uint8 i2cAddr, uint8 i2cCmd, uint8 *i2cVal)
Henter en I2C pakke.

Private metoder

- static uint8 `i2c_tx` (uint8 i2cAddr, uint8 i2cCmd, uint8 i2cVal)
Buffer til afsendelse af data.
- static uint8 `i2c_rx` (uint8 i2cRxAddr, uint8 *i2cRxCmd, uint8 *i2cRxVal)
Buffer til modtagelse af data.

3.3.1 Detaljeret beskrivelse

`I2C` class.

Håndter kommunikation via I2C-busset.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

3.3.2 Dokumentation af medlemsfunktioner

3.3.2.1 void `i2c_getPacket` (uint8 `i2cAddr`, uint8 `i2cCmd`, uint8 * `i2cVal`)

Henter en `I2C` pakke.

Metoden henter en `I2C` data pakke via I2C-busset fra den defineret adresse med den modtaget kommande og lager den på den modtaget værdi pointer.

Parametre

in	<code>i2cAddr</code>	<code>I2C</code> adresse på modtager.
in	<code>i2cCmd</code>	Kommando til modtager.
out	<code>i2cVal</code>	Pointer til variabel hvor den hentet værdi skal lagers.

Returnerer

Status på kommunikation.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Defineret på linje 99 i filen `i2c.c`.

Indeholder referencer til `i2c_rx()`, `I2C_STS_CMD_DONE`, `i2c_tx()`, `LCD::lcd_newline()` og `LED::setLed()`.

Refereret til af `Handler::handler()`.

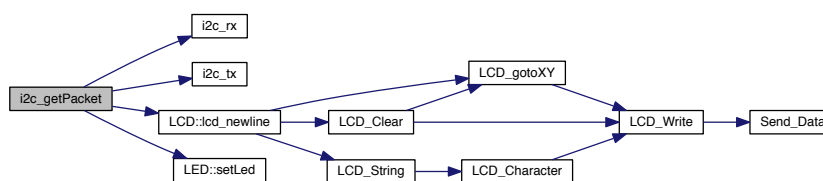
```
100 {  
101     uint8 status;  
102     uint8 i2cTxSTS;  
103     uint8 i2cRxCmd;  
104     char lcd[12];
```

```

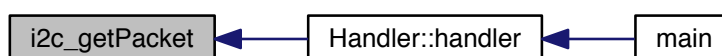
105
106 i2cTxSTS = i2c_tx(i2cAddr, i2cCmd, *i2cVal);
107 if(i2cTxSTS == I2C_STS_CMD_DONE)
108 {
109     status = 1;
110     setLed(0,0,1,50);
111 }
112 else
113 {
114     status = 0;
115     setLed(1,0,0,50);
116 }
117 sprintf(lcd, ">I %1.1x %2.2x %2.2x %1.1d", (int)i2cAddr, (int)i2cCmd, (int)*i2cVal, status);
118 lcd_newline(lcd);
119
120 DEBUG_PutString("I>: addr: ");
121 DEBUG_PutHexByte(i2cAddr);
122 DEBUG_PutString(" cmd: ");
123 DEBUG_PutHexByte(i2cCmd);
124 DEBUG_PutString(" val: ");
125 DEBUG_PutHexByte(*i2cVal);
126 DEBUG_PutCRLF();
127
128 i2c_rx(i2cAddr, &i2cRxCmd, i2cVal);
129 if(i2cRxCmd == i2cCmd)
130 {
131     status = 1;
132     setLed(0,1,0,50);
133 }
134 else
135 {
136     status = 0;
137     setLed(1,0,0,50);
138 }
139 setLed(0,0,0,50);
140
141 sprintf(lcd, ">I %1.1x %2.2x %2.2x %1.1d", (int)i2cAddr, (int)i2cCmd, (int)i2cVal, status);
142 lcd_newline(lcd);
143 DEBUG_PutString(">I: addr: ");
144 DEBUG_PutHexByte(i2cAddr);
145 DEBUG_PutString(" cmd: ");
146 DEBUG_PutHexByte(i2cCmd);
147 DEBUG_PutString(" val: ");
148 DEBUG_PutHexByte(*i2cVal);
149 DEBUG_PutCRLF();
150 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



3.3.2.2 void i2c_init (void)

Initialiser I2C modulet.

Initailiser I2C komponent på PSoC'en.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Defineret på linje 46 i filen i2c.c.

Refereret til af main().

```
47 {  
48     I2CM_Start();  
49 }
```

Her er kalder-grafen for denne funktion:



3.3.2.3 uint8 i2c_rx (uint8 i2cRxAddr, uint8 * i2cRxCmd, uint8 * i2cRxVal) [private]

Buffer til modtagelse af data.

Henter en I2C pakke.

En buffer der indeholder de data pakker der skal modtagelse over I2C-busset.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Metoden henter en I2C data pakke via I2C-busset fra den defineret adresse med den modtaget kommande og lager den på den modtaget værdi pointer.

Parametre

in	<i>i2cRxAddr</i>	I2C adresse på modtager.
in	<i>i2cRxCmd</i>	Kommando til modtager.
out	<i>i2cRxVal</i>	Pointer til variabel hvor den hentet værdi skal lagers.

Returnerer

Status på kommunikation.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Defineret på linje 205 i filen i2c.c.

Indeholder referencer til I2C_BUFFER_SIZE, I2C_PACKET_CMD_POS, I2C_PACKET_EOP, I2C_PACKET_EOP_POS, I2C_PACKET_SIZE, I2C_PACKET_SOP, I2C_PACKET_SOP_POS, I2C_PACKET_VAL_POS og I2C_STS_CMD_FAIL.

Refereret til af i2c_getPacket().

```

206 {
207     uint8 i2cRxStatus = I2C_STS_CMD_FAIL;
208     uint8 i2cRxData[I2C_PACKET_SIZE];
209
210     (void) I2CM_I2CMasterReadBuf(i2cRxAddr, i2cRxData, I2C_PACKET_SIZE,
I2CM_I2C_MODE_COMPLETE_XFER);
211     while (0u == (I2CM_I2CMasterStatus() & I2CM_I2C_MSTAT_RD_CMPLT))
212     {
213     }
214     if (0u == (I2CM_I2C_MSTAT_ERR_XFER & I2CM_I2CMasterStatus()))
215     {
216         if ((I2CM_I2CMasterGetReadBufSize() == I2C_BUFFER_SIZE))
217         {
218             if ((i2cRxData[I2C_PACKET_SOP_POS] == I2C_PACKET_SOP) && (i2cRxData[
I2C_PACKET_EOP_POS] == I2C_PACKET_EOP))
219             {
220                 *i2cRxCmd = i2cRxData[I2C_PACKET_CMD_POS];
221                 *i2cRxVal = i2cRxData[I2C_PACKET_VAL_POS];
222                 i2cRxStatus = i2cRxData[I2C_PACKET_CMD_POS];
223             }
224         }
225     }
226     (void) I2CM_I2CMasterClearStatus();
227
228     return i2cRxStatus;
229 }

```

Her er kalder-grafen for denne funktion:



3.3.2.4 void i2c_setPacket (uint8 i2cAddr, uint8 i2cCmd, uint8 i2cVal)

Sender en I2C pakke.

Metoden sender en I2C data pakke via I2C-busset til den defineret adresse med den modtaget kommande og tilhørende værdi.

Parametre

in	<i>i2cAddr</i>	I2C adresse på modtager.
in	<i>i2cCmd</i>	Kommando til modtager.
in	<i>i2cVal</i>	Værdi til modtager.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

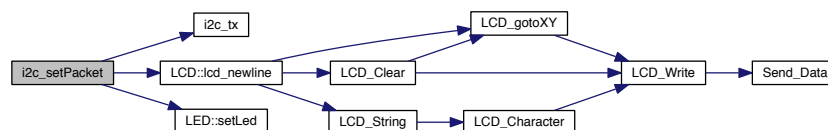
Defineret på linje 61 i filen i2c.c.

Indeholder referencer til I2C_STS_CMD_DONE, i2c_tx(), LCD::lcd_newline() og LED::setLed().

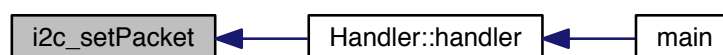
Refereret til af Handler::handler().

```
62 {
63     uint8 status;
64     char lcd[12];
65     if(i2c_tx(i2cAddr, i2cCmd, i2cVal) == I2C_STS_CMD_DONE)
66     {
67         status = 1;
68         setLed(0,0,1,50);
69     }
70     else
71     {
72         status = 0;
73         setLed(1,0,0,50);
74     }
75     sprintf(lcd, "I>%2.1x %2.2x %2.2x %1.1d", (int)i2cAddr, (int)i2cCmd, (int)i2cVal, status);
76     lcd_newline(lcd);
77
78     DEBUG_PutString("I>: addr: ");
79     DEBUG_PutHexByte(i2cAddr);
80     DEBUG_PutString(" cmd: ");
81     DEBUG_PutHexByte(i2cCmd);
82     DEBUG_PutString(" val: ");
83     DEBUG_PutHexByte(i2cVal);
84     DEBUG_PutCRLF();
85     setLed(0,0,0,50);
86 }
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



3.3.2.5 uint8 i2c_tx (uint8 i2cAddr, uint8 i2cCmd, uint8 i2cVal) [private]

Buffer til afsendelse af data.

Sender en I2C pakke.

En buffer der indeholder de data pakker der skal sende over I2C-busset.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Metoden sender en I2C data pakke via I2C-busset til den defineret adresse med den modtaget kommande og tilhørende værdi.

Parametre

in	i2cAddr	I2C adresse på modtager.
in	i2cCmd	Kommando til modtager.
in	i2cVal	Værdi til modtager.

Returnerer

Status på kommunikation.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

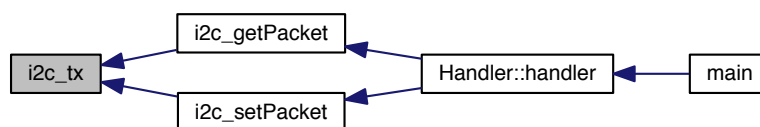
Defineret på linje 168 i filen i2c.c.

Indeholder referencer til I2C_BUFFER_SIZE, I2C_PACKET_CMD_POS, I2C_PACKET_EOP, I2C_PACKET_EOP_POS, I2C_PACKET_SIZE, I2C_PACKET_SOP, I2C_PACKET_SOP_POS, I2C_PACKET_VAL_POS, I2C_STS_CMD_DONE og I2C_STS_CMD_FAIL.

Refereret til af i2c_getPacket() og i2c_setPacket().

```
169 {
170     uint8 i2cTxStatus = I2C_STS_CMD_FAIL;
171     uint8 i2cTxData[I2C_PACKET_SIZE];
172
173     i2cTxData[I2C_PACKET_SOP_POS] = I2C_PACKET_SOP;
174     i2cTxData[I2C_PACKET_CMD_POS] = i2cCmd;
175     i2cTxData[I2C_PACKET_VAL_POS] = i2cVal;
176     i2cTxData[I2C_PACKET_EOP_POS] = I2C_PACKET_EOP;
177
178     (void) I2CM_I2CMasterWriteBuf(i2cAddr, i2cTxData, I2C_PACKET_SIZE,
179     I2CM_I2C_MODE_COMPLETE_XFER);
179     while (0u == (I2CM_I2CMasterStatus() & I2CM_I2C_MSTAT_WR_CMPLT))
180     {
181     }
182     if (0u == (I2CM_I2C_MSTAT_ERR_XFER & I2CM_I2CMasterStatus()))
183     {
184         if (I2CM_I2CMasterGetWriteBufSize() == I2C_BUFFER_SIZE)
185         {
186             i2cTxStatus = I2C_STS_CMD_DONE;
187         }
188     }
189     (void) I2CM_I2CMasterClearStatus();
190
191     return i2cTxStatus;
192 }
```

Her er kalder-grafen for denne funktion:



Dokumentationen for denne klasse blev genereret ud fra filerne:

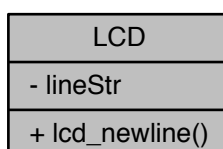
- [i2c.h](#)
- [i2c.c](#)

3.4 LCD Klasse-reference

[LCD](#) class.

```
#include <lcd.h>
```

Samarbejdsdiagram for LCD:



Offentlige metoder

- void [lcd_newline](#) (char *characters)
Udskriver tekst på Nokia 5110 [LCD](#).

Private attributter

- char [lineStr](#) [6][12]
Char array der indholder tekst.

3.4.1 Detaljeret beskrivelse

LCD class.

Sender tekst til Nokia5110LCD skærmen via dens eksterne kode.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

3.4.2 Dokumentation af medlemsfunktioner

3.4.2.1 void lcd_newline (char * characters)

Udskriver tekst på Nokia 5110 LCD.

Metoden bruges til at skrive en ny linje nederst på Nokia 5110 LCD skærmen, den husker på- og flytter de forhen-værende linjer en linje op, når der indættes en ny.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Defineret på linje 36 i filen lcd.c.

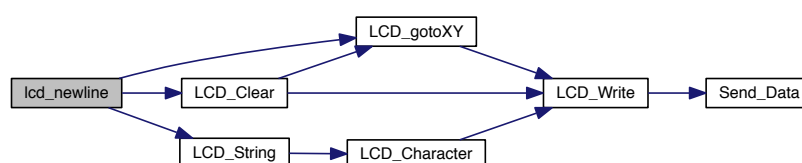
Indeholder referencer til LCD_Clear(), LCD_gotoXY(), LCD_String() og lineStr.

Refereret til af SPI::CY_ISR(), I2C::i2c_getPacket(), I2C::i2c_setPacket() og main().

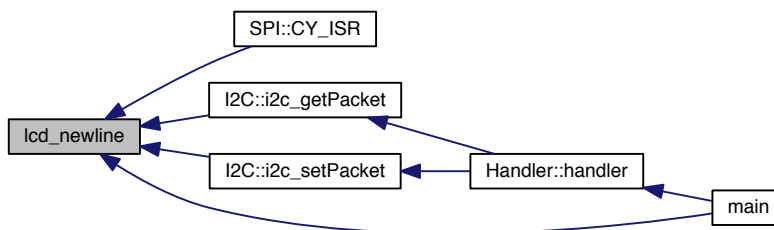
```

37 {
38   int i;
39
40   for(i = 0; i < 5; i++)
41   {
42     strncpy(lineStr[i], lineStr[i+1], 12);
43   }
44
45   strcpy(lineStr[5], characters);
46
47   LCD_Clear();
48   for(i = 0; i < 6; i++)
49   {
50     LCD_gotoXY(0, i);
51     LCD_String(lineStr[i]);
52   }
53 }
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



3.4.3 Felt-dokumentation

3.4.3.1 `char lineStr[6][12] [private]`

Char array der indholder tekst.

Arrayet er et matrix array med 6 arryes med 12 pladser, det bruges til at indeholde de 6 linjer tekst der kan udskrives på Nokia 5110 LCD skærmen.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Defineret på linje 22 i filen `lcd.c`.

Refereret til af `lcd_newline()`.

Dokumentationen for denne klasse blev genereret ud fra filerne:

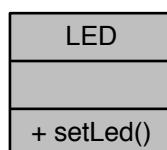
- [lcd.h](#)
- [lcd.c](#)

3.5 LED Klasse-reference

[LED](#) class.

```
#include <led.h>
```

Samarbejdsdiagram for LED:



Offentlige metoder

- void `setLed` (uint8 red, uint8 green, uint8 blue, uint8 delay)
Sætter den defineret farve og angivet delay.

3.5.1 Detaljeret beskrivelse

`LED` class.

Håndtere PSoC'ens røde, grønne og blå led

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

3.5.2 Dokumentation af medlemsfunktioner

3.5.2.1 void `setLed` (uint8 *red*, uint8 *green*, uint8 *blue*, uint8 *delay*)

Sætter den defineret farve og angivet delay.

Metoden sætter den/de valgte farver og venter i det angivet delay.

Parametre

in	<i>red</i>	Tænder/slukker den røde led.
in	<i>green</i>	Tænder/slukker den grønne led.
in	<i>blue</i>	Tænder/slukker den blå led.
in	<i>delay</i>	Tid i microsekunder til delay.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

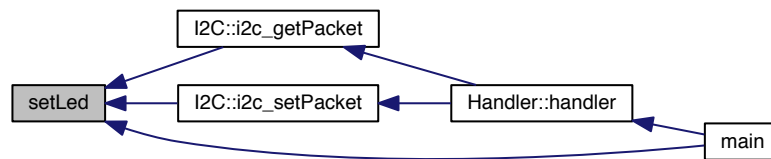
Defineret på linje 24 i filen `led.c`.

Indeholder referencer til `LED_OFF` og `LED_ON`.

Refereret til af `I2C::i2c_getPacket()`, `I2C::i2c_setPacket()` og `main()`.

```
25 {  
26   red ? LED_RED_Write(LED_ON) : LED_RED_Write(LED_OFF);  
27   green ? LED_GREEN_Write(LED_ON) : LED_GREEN_Write(LED_OFF);  
28   blue ? LED_BLUE_Write(LED_ON) : LED_BLUE_Write(LED_OFF);  
29  
30   CyDelay(delay);  
31 }
```

Her er kalder-grafen for denne funktion:



Dokumentationen for denne klasse blev genereret ud fra filerne:

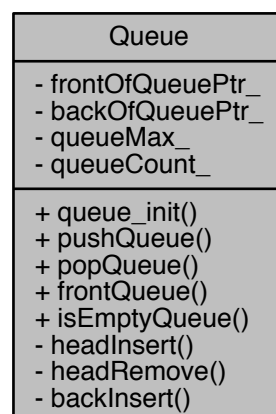
- [led.h](#)
- [led.c](#)

3.6 Queue Klasse-reference

[Queue](#) class.

```
#include <queue.h>
```

Samarbejdsdiagram for Queue:



Offentlige metoder

- void `queue_init` (uint8 queueMaxSize)
Initialiser `Queue` modulet.
- void `pushQueue` (const struct `Action` data)
Indsætter et element i køen.
- void `popQueue` ()
Fjerner et element i køen.
- struct `Action` `frontQueue` ()
Viser et element fra køen.
- uint8 `isEmptyQueue` ()
Retuner status af køen.

Private metoder

- void `headInsert` (struct `Node` **headPtr, const struct `Action` data)
Indsætter forreste i listen.
- void `headRemove` (struct `Node` **headPtr)
Fjerner fra listen.
- void `backInsert` (struct `Node` **backPtr, const struct `Action` data)
Indsætter bagerst i listen.

Statiske, private attributter

- static struct `Node` * `frontOfQueuePtr_`
Pointer til foreste element i køen.
- static struct `Node` * `backOfQueuePtr_`
Pointer til bagerste element i køen.
- static uint8 `queueMax_`
Køens max.
- static uint8 `queueCount_`
Kø element tæller.

3.6.1 Detaljeret beskrivelse

`Queue` class.

En FIFO kø der er opbygget af en single linket liste.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

3.6.2 Dokumentation af medlemsfunktioner

3.6.2.1 void backInsert (struct `Node` ** `backPtr`, const struct `Action` `data`) [private]

Indsætter bagerst i listen.

Indsætter det angivet element bagerst i den underlægende linked liste.

Parametre

in	<i>backPtr</i>	Pointer til det bagerste element i listen.
in	<i>data</i>	Data der skal indsættes i listen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 246 i filen queue.c.

Indeholder referencer til Node::data_ og Node::next_.

```

247 {
248     if(*backPtr == NULL)
249     {
250         return;
251     }
252
253     struct Node* next = (*backPtr)->next_;
254     struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
255     temp->data_ = data;
256     temp->next_ = next;
257     (*backPtr)->next_ = temp;
258 }
```

3.6.2.2 struct Action frontQueue (void)

Viser et element fra køen.

Viser det foreste element i FIFO køen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 168 i filen queue.c.

Indeholder referencer til Node::data_.

Refereret til af main().

```

169 {
170     DEBUG_PutString("Q=: count: ");
171     DEBUG_PutHexByte(queueCount_);
172     DEBUG_PutCRLF();
173     return frontOfQueuePtr_->data_;
174 }
```

Her er kalder-grafen for denne funktion:



3.6.2.3 void headInsert (struct Node ** headPtr, const struct Action data) [private]

Indsætter forreste i listen.

Indsætter det angivet element forreste i den underlægende linked liste.

Parametre

in	<i>headPtr</i>	Pointer til det foreste element i listen.
in	<i>data</i>	Data der skal indsættes i listen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 204 i filen queue.c.

Indeholder referencer til Node::data_ og Node::next_.

```
205 {
206     struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
207     if(temp == NULL)
208     {
209         return;
210     }
211
212     temp->data_ = data;
213     temp->next_ = NULL;
214
215     *headPtr = temp;
216 }
```

3.6.2.4 void headRemove (struct Node ** headPtr) [private]

Fjerner fra listen.

Fjerner det forreste element i den underlæggende linked liste

Parametre

in	<i>headPtr</i>	Pointer til det forreste element i listen.
----	----------------	--

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 226 i filen queue.c.

Indeholder referencer til Node::next_.

```
227 {
228     if(headPtr != NULL)
229     {
230         struct Node* condemned;
231         condemned = *headPtr;
232         *headPtr = (*headPtr)->next_;
233         free(condemned);
234     }
235 }
```

3.6.2.5 uint8 isEmptyQueue (void)

Retuner status af køen.

Kontrollere om køen er tom.

Forfatter

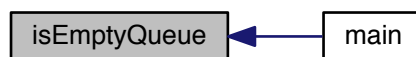
Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 183 i filen queue.c.

Refereret til af main().

```
184 {  
185     if(frontOfQueuePtr_ == NULL)  
186     {  
187         return 1;  
188     }  
189     else  
190     {  
191         return 0;  
192     }  
193 }
```

Her er kalder-grafen for denne funktion:



3.6.2.6 void popQueue (void)

Fjerner et element i køen.

Fjerner det foreste element i FIFO køen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 147 i filen queue.c.

Indeholder referencer til headRemove() og isEmptyQueue().

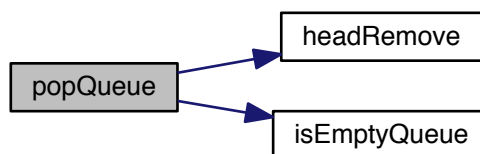
Refereret til af main().

```

148 {
149     headRemove(&frontOfQueuePtr_);
150     queueCount_--;
151     if(isEmptyQueue() == 1)
152     {
153         backOfQueuePtr_ = NULL;
154     }
155     DEBUG_PutString("-Q: count: ");
156     DEBUG_PutHexByte(queueCount_);
157     DEBUG_PutCRLF();
158     DEBUG_PutCRLF();
159 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



3.6.2.7 void pushQueue (const struct Action data)

Indsætter et element i køen.

Indsætter det angivne element bagerst i FIFO køen.

Parametre

in	data	Data der skal indsættes i køen.
----	------	---------------------------------

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

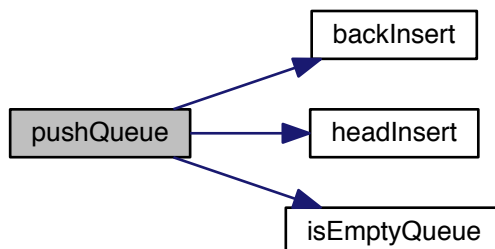
Defineret på linje 105 i filen queue.c.

Indeholder referencer til backInsert(), Action::cmd, headInsert(), isEmptyQueue(), Node::next_ og Action::val.

Refereret til af SPI::CY_ISR().

```
106 {
107     if(queueCount_<queueMax_)
108     {
109         if(isEmptyQueue() != 1)
110         {
111             backInsert(&backOfQueuePtr_, data);
112             backOfQueuePtr_ = backOfQueuePtr_>next_;
113             queueCount_++;
114         }
115     }
116     else
117     {
118         headInsert(&frontOfQueuePtr_, data);
119         backOfQueuePtr_ = frontOfQueuePtr_;
120         queueCount_++;
121     }
122     DEBUG_PutString("Q+: count: ");
123     DEBUG_PutHexByte(queueCount_);
124     DEBUG_PutString(" cmd: ");
125     DEBUG_PutHexByte(data.cmd);
126     DEBUG_PutString(" val: ");
127     DEBUG_PutHexByte(data.val);
128     DEBUG_PutCRLF();
129 }
130 else
131 {
132     DEBUG_PutString("Q~: ERROR! Queue FULL!!! count: ");
133     DEBUG_PutHexByte(queueCount_);
134     DEBUG_PutCRLF();
135     DEBUG_PutCRLF();
136 }
137 }
138 }
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



3.6.2.8 void queue_init (uint8 queueMaxSize)

Initialiser `Queue` modulet.

Initialiser køen med den ønsket max størrelse.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 87 i filen queue.c.

Indeholder referencer til `Node::next_`.

Refereret til af `main()`.

```
88 {
89     frontOfQueuePtr_ = NULL;
90     frontOfQueuePtr_>next_ = NULL;
91     backOfQueuePtr_ = NULL;
92     backOfQueuePtr_>next_ = NULL;
93     queueMax_ = queueMaxSize;
94     queueCount_ = 0;
95 }
```

Her er kalder-grafen for denne funktion:



3.6.3 Felt-dokumentation

3.6.3.1 struct Node* backOfQueuePtr_ [static],[private]

Pointer til bagerste element i køen.

En `Node` pointer der indeholder adressen på det bagerste elementet i køen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 47 i filen queue.c.

3.6.3.2 struct Node* frontOfQueuePtr_ [static],[private]

Pointer til foreste element i køen.

En [Node](#) pointer der indeholder adressen på det foreste elementet i køen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 38 i filen queue.c.

3.6.3.3 uint8 queueCount_ [static],[private]

Kø element tæller.

Bruges til at tælle hvor mange elementer der er i køen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 65 i filen queue.c.

3.6.3.4 uint8 queueMax_ [static],[private]

Køens max.

Laver ved initialisering der ønsket antal for max elementer i køen

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 56 i filen queue.c.

Dokumentationen for denne klasse blev genereret ud fra filerne:

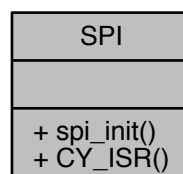
- [queue.h](#)
- [queue.c](#)

3.7 SPI Klasse-reference

[SPI](#) class.

```
#include <spi.h>
```

Samarbejdsdiagram for SPI:



Offentlige metoder

- void `spi_init()`
Initialiser SPI modulet.
- `CY_ISR` (`isr_spi_rx`)
Modtager kald fra SPI-busset.

3.7.1 Detaljeret beskrivelse

`SPI` class.

Håndter kommunikation via SPI-busset.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

3.7.2 Dokumentation af medlemsfunktioner

3.7.2.1 `CY_ISR` (`isr_spi_rx`)

Modtager kald fra SPI-busset.

En "Interrupt Service Routine(ISR)" der aktiveres ved modtagelse af kald via SPI-busset, det modtagne data behandles og håndteres.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Defineret på linje 41 i filen `spi.c`.

Indeholder referencer til `DataMaster::bVal`, `Action::cmd`, `CMD_GET_BLUE_VAL`, `CMD_GET_GREEN_VAL`, `CMD_GET_RED_VAL`, `CMD_GET_X_POS`, `CMD_GET_Y_POS`, `CMD_GET_Z_POS`, `dataMaster`, `DataMaster::gVal`, `LCD::lcd_newline()`, `Queue::pushQueue()`, `DataMaster::rVal`, `SPI_PACKET_DATA_POS`, `SPI_PACKET_SIZE`, `Action::val`, `DataMaster::xVal`, `DataMaster::yVal` og `DataMaster::zVal`.

```

42 {
43     SPIS_DisableInt();
44     char lcd[12];
45     uint16 spiRxBuffer[SPI_PACKET_SIZE];
46     uint16 spiTxBuffer[SPI_PACKET_SIZE];
47     struct Action spiRxAction;
48     DEBUG_PutString("SPI!");
49     DEBUG_PutCRLF();
50
51     while(SPIS_SpiUartGetRxBufferSize() > 0)
52     {
53         spiRxBuffer[SPI_PACKET_DATA_POS] = SPIS_SpiUartReadRxData();
54         spiRxAction.val = spiRxBuffer[SPI_PACKET_DATA_POS] & 0xff;
55         spiRxAction.cmd = (spiRxBuffer[SPI_PACKET_DATA_POS] >> 8);
56
57         if(spiRxBuffer[SPI_PACKET_DATA_POS] == 0xBADA)
58         {
59             sprintf(lcd, "S> %x", spiTxBuffer[SPI_PACKET_DATA_POS]);
60             lcd_newline(lcd);
61
62             DEBUG_PutString("S>: val: ");
63             DEBUG_PutHexByte(spiTxBuffer[SPI_PACKET_DATA_POS]);
64             DEBUG_PutCRLF();
65         }

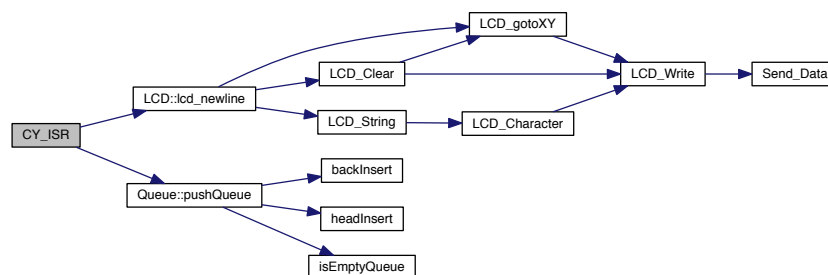
```

```

66     else
67     {
68         sprintf(lcd, ">S %4x %2x", (int)spiRxAction.cmd, (int)spiRxAction.val);
69         lcd_newline(lcd);
70
71         DEBUG_PutString(">S: cmd: ");
72         DEBUG_PutHexByte(spiRxAction.cmd);
73         DEBUG_PutString(" val: ");
74         DEBUG_PutHexByte(spiRxAction.val);
75         DEBUG_PutCRLF();
76         DEBUG_PutCRLF();
77
78         switch(spiRxAction.cmd) {
79             case CMD_GET_X_POS :
80                 SPIS_SpiUartClearTxBuffer();
81                 spiTxBuffer[SPI_PACKET_DATA_POS] = (uint16)
dataMaster.xVal;
82                 SPIS_SpiUartPutArray(spiTxBuffer, SPI_PACKET_SIZE);
83                 break;
84             case CMD_GET_Y_POS :
85                 SPIS_SpiUartClearTxBuffer();
86                 spiTxBuffer[SPI_PACKET_DATA_POS] = (uint16)
dataMaster.yVal;
87                 SPIS_SpiUartPutArray(spiTxBuffer, SPI_PACKET_SIZE);
88                 break;
89             case CMD_GET_Z_POS :
90                 SPIS_SpiUartClearTxBuffer();
91                 spiTxBuffer[SPI_PACKET_DATA_POS] = (uint16)
dataMaster.zVal;
92                 SPIS_SpiUartPutArray(spiTxBuffer, SPI_PACKET_SIZE);
93                 break;
94             case CMD_GET_RED_VAL :
95                 SPIS_SpiUartClearTxBuffer();
96                 spiTxBuffer[SPI_PACKET_DATA_POS] = (uint16)
dataMaster.rVal;
97                 SPIS_SpiUartPutArray(spiTxBuffer, SPI_PACKET_SIZE);
98                 break;
99             case CMD_GET_GREEN_VAL :
100                 SPIS_SpiUartClearTxBuffer();
101                 spiTxBuffer[SPI_PACKET_DATA_POS] = (uint16)
dataMaster.gVal;
102                 SPIS_SpiUartPutArray(spiTxBuffer, SPI_PACKET_SIZE);
103                 break;
104             case CMD_GET_BLUE_VAL :
105                 SPIS_SpiUartClearTxBuffer();
106                 spiTxBuffer[SPI_PACKET_DATA_POS] = (uint16)
dataMaster.bVal;
107                 SPIS_SpiUartPutArray(spiTxBuffer, SPI_PACKET_SIZE);
108                 break;
109             default :
110                 pushQueue(spiRxAction);
111                 break;
112         }
113     }
114 }
115
116 SPIS_ClearRxInterruptSource (SPIS_GetRxInterruptSource());
117 SPIS_EnableInt();
118 }

```

Her er kald-grafen for denne funktion:



3.7.2.2 void spi_init (void)

Initialiser [SPI](#) modulet.

Initialiser [SPI](#) komponent på PSoC'en og sætter "Custom Interrupt Handler".

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

Defineret på linje 25 i filen spi.c.

Refereret til af main().

```
26 {  
27     SPIS_SpiUartClearTxBuffer();  
28     SPIS_SpiUartClearRxBuffer();  
29     SPIS_SetCustomInterruptHandler(isr_spi_rx);  
30  
31     SPIS_Start();  
32 }
```

Her er kalder-grafen for denne funktion:



Dokumentationen for denne klasse blev genereret ud fra filerne:

- [spi.h](#)
- [spi.c](#)

4 Fil-dokumentation

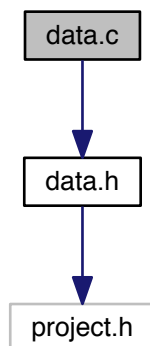
4.1 cyapicalcallbacks.h filreference

4.2 data.c filreference

[Data](#) modul.

```
#include "data.h"
```

Inklusions-afhængighedsgraf for data.c:



4.2.1 Detaljeret beskrivelse

Data modul.

Indeholder data hentet fra PSoC-XY, -Z og -Sensor.

Forfatter

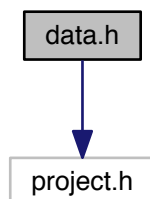
Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.3 data.h filreference

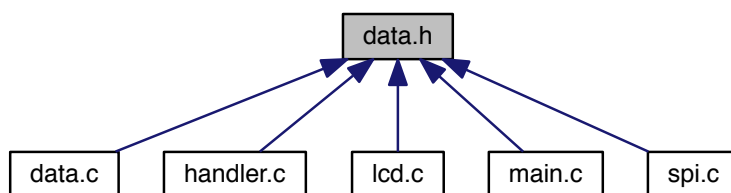
Data modul.

```
#include <project.h>
```

Inklusions-afhængighedsgraf for data.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



Datastrukturer

- struct [DataMaster](#)
[Data struct. Mere...](#)

Funktioner

- void [data_init](#) (void)

Variable

- struct [DataMaster](#) [dataMaster](#)

4.3.1 Detaljeret beskrivelse

[Data](#) modul.

Indeholder data hentet fra PSoC-XY, -Z og -Sensor.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.3.2 Datastruktur-documentation

4.3.2.1 struct DataMaster

[Data](#) struct.

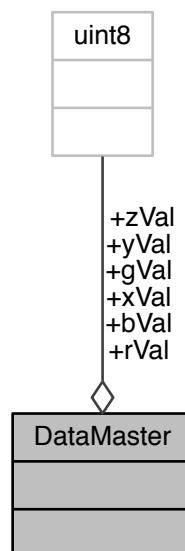
En data struct der indeholder de sidst kendte værdier fra PSoC-XY -Z og -Sensor.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 34 i filen data.h.

Samarbejdsdiagram for DataMaster:



Data-felter

uint8	bVal	Værdi for sidst kendte B niveau
uint8	gVal	Værdi for sidst kendte G niveau
uint8	rVal	Værdi for sidst kendte R niveau
uint8	xVal	Værdi for sidst kendte X position
uint8	yVal	Værdi for sidst kendte Y position
uint8	zVal	Værdi for sidst kendte Z position

4.3.3 Funktions-dokumentation

4.3.3.1 void data_init (void)

4.3.4 Variabel-dokumentation

4.3.4.1 struct DataMaster dataMaster

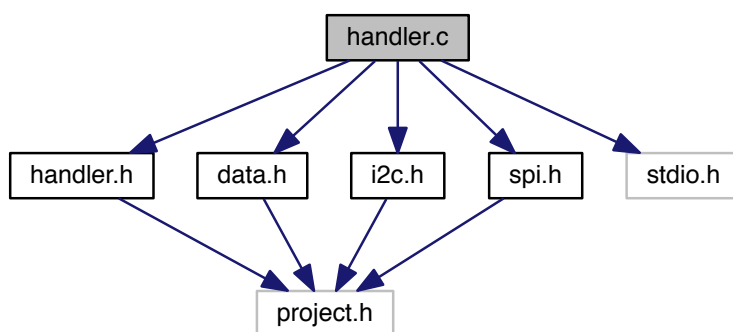
Refereret til af SPI::CY_ISR(), Data::data_init() og Handler::handler().

4.4 handler.c filreference

Handler modul.

```
#include "handler.h"  
#include "data.h"  
#include "i2c.h"  
#include "spi.h"  
#include <stdio.h>
```

Inklusions-afhængighedsgraf for handler.c:



4.4.1 Detaljeret beskrivelse

Handler modul.

Håndtere indkommende kommandoer med tilhørende værdier.

Forfatter

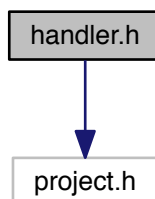
Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.5 handler.h filreference

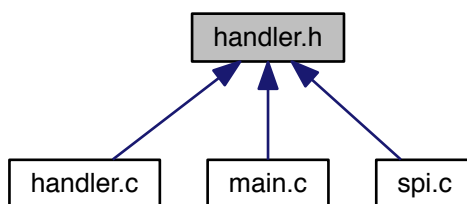
Handler modul.

```
#include <project.h>
```

Inklusions-afhængighedsgraf for handler.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



#Defines

- `#define CMD_SET_X_POS` (0x10u)
- `#define CMD_SET_Y_POS` (0x11u)
- `#define CMD_GET_X_POS` (0x12u)
- `#define CMD_GET_Y_POS` (0x13u)
- `#define CMD_GET_X_MAX` (0x14u)
- `#define CMD_GET_Y_MAX` (0x15u)
- `#define CMD_X_STP` (0x16u)
- `#define CMD_Y_STP` (0x17u)
- `#define CMD_X_CAL` (0x18u)
- `#define CMD_Y_CAL` (0x19u)
- `#define CMD_SET_Z_POS` (0x20u)
- `#define CMD_GET_Z_POS` (0x21u)
- `#define CMD_GET_Z_MAX` (0x22u)
- `#define CMD_Z_STP` (0x23u)
- `#define CMD_Z_CAL` (0x24u)
- `#define CMD_SET_RED_VAL` (0x30u)
- `#define CMD_SET_GREEN_VAL` (0x31u)
- `#define CMD_SET_BLUE_VAL` (0x32u)
- `#define CMD_SET_LUMEN_VAL` (0x33u)
- `#define CMD_SET_POWER_STS` (0x34u)
- `#define CMD_GET_RED_VAL` (0x35u)
- `#define CMD_GET_GREEN_VAL` (0x36u)
- `#define CMD_GET_BLUE_VAL` (0x37u)
- `#define CMD_GET_LUMEN_VAL` (0x38u)
- `#define CMD_GET_POWER_STS` (0x39u)
- `#define CMD_SET_DISTANCE_STS` (0x40u)
- `#define CMD_SET_MOVEMENT_STS` (0x41u)
- `#define CMD_GET_DISTANCE_STS` (0x42u)
- `#define CMD_GET_MOVEMENT_STS` (0x43u)
- `#define CMD_DISTANCE_ALRT` (0x44u)
- `#define CMD_MOVEMENT_ALRT` (0x45u)

Funktioner

- void `handler` (uint8 cmd, uint8 val)

4.5.1 Detaljeret beskrivelse

[Handler](#) modul.

Håndtere indkommende kommandoer med tilhørende værdier.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.5.2 #Define-dokumentation

4.5.2.1 #define CMD_DISTANCE_ALERT (0x44u)

Defineret på linje 65 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.2 #define CMD_GET_BLUE_VAL (0x37u)

Defineret på linje 58 i filen handler.h.

Refereret til af SPI::CY_ISR() og Handler::handler().

4.5.2.3 #define CMD_GET_DISTANCE_STS (0x42u)

Defineret på linje 63 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.4 #define CMD_GET_GREEN_VAL (0x36u)

Defineret på linje 57 i filen handler.h.

Refereret til af SPI::CY_ISR() og Handler::handler().

4.5.2.5 #define CMD_GET_LUMEN_VAL (0x38u)

Defineret på linje 59 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.6 #define CMD_GET_MOVEMENT_STS (0x43u)

Defineret på linje 64 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.7 #define CMD_GET_POWER_STS (0x39u)

Defineret på linje 60 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.8 #define CMD_GET_RED_VAL (0x35u)

Defineret på linje 56 i filen handler.h.

Refereret til af SPI::CY_ISR() og Handler::handler().

4.5.2.9 #define CMD_GET_X_MAX (0x14u)

Defineret på linje 40 i filen handler.h.

4.5.2.10 #define CMD_GET_X_POS (0x12u)

Defineret på linje 38 i filen handler.h.

Refereret til af SPI::CY_ISR() og Handler::handler().

4.5.2.11 #define CMD_GET_Y_MAX (0x15u)

Defineret på linje 41 i filen handler.h.

4.5.2.12 #define CMD_GET_Y_POS (0x13u)

Defineret på linje 39 i filen handler.h.

Refereret til af SPI::CY_ISR() og Handler::handler().

4.5.2.13 #define CMD_GET_Z_MAX (0x22u)

Defineret på linje 48 i filen handler.h.

4.5.2.14 #define CMD_GET_Z_POS (0x21u)

Defineret på linje 47 i filen handler.h.

Refereret til af SPI::CY_ISR() og Handler::handler().

4.5.2.15 #define CMD_MOVEMENT_ALRT (0x45u)

Defineret på linje 66 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.16 #define CMD_SET_BLUE_VAL (0x32u)

Defineret på linje 53 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.17 #define CMD_SET_DISTANCE_STS (0x40u)

Defineret på linje 61 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.18 #define CMD_SET_GREEN_VAL (0x31u)

Defineret på linje 52 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.19 #define CMD_SET_LUMEN_VAL (0x33u)

Defineret på linje 54 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.20 #define CMD_SET_MOVEMENT_STS (0x41u)

Defineret på linje 62 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.21 #define CMD_SET_POWER_STS (0x34u)

Defineret på linje 55 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.22 #define CMD_SET_RED_VAL (0x30u)

Defineret på linje 51 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.23 #define CMD_SET_X_POS (0x10u)

Defineret på linje 36 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.24 #define CMD_SET_Y_POS (0x11u)

Defineret på linje 37 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.25 #define CMD_SET_Z_POS (0x20u)

Defineret på linje 46 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.26 #define CMD_X_CAL (0x18u)

Defineret på linje 44 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.27 #define CMD_X_STP (0x16u)

Defineret på linje 42 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.28 #define CMD_Y_CAL (0x19u)

Defineret på linje 45 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.29 #define CMD_Y_STP (0x17u)

Defineret på linje 43 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.30 #define CMD_Z_CAL (0x24u)

Defineret på linje 50 i filen handler.h.

Refereret til af Handler::handler().

4.5.2.31 #define CMD_Z_STP (0x23u)

Defineret på linje 49 i filen handler.h.

Refereret til af Handler::handler().

4.5.3 Funktions-dokumentation

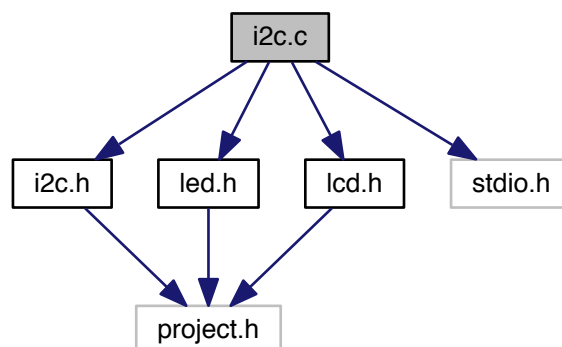
4.5.3.1 void handler (uint8 cmd, uint8 val)

4.6 i2c.c filreference

I2C modul.

```
#include "i2c.h"  
#include "led.h"  
#include "lcd.h"  
#include <stdio.h>
```

Inklusions-afhængighedsgraf for i2c.c:



4.6.1 Detaljeret beskrivelse

I2C modul.

Håndter kommunikation via I2C-busset

Forfatter

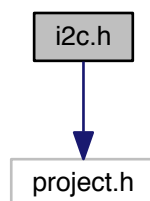
Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.7 i2c.h filreference

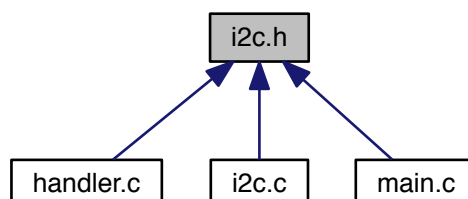
I2C modul.

```
#include <project.h>
```

Inklusions-afhængighedsgraf for i2c.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



#Defines

- #define `PSoC_XY` (0x08u)
- #define `PSoC_Z` (0x09u)
- #define `PSoC_Sensor` (0x10u)
- #define `I2C_BUFFER_SIZE` (4u)
- #define `I2C_PACKET_SIZE` (4u)
- #define `I2C_PACKET_SOP_POS` (0u)
- #define `I2C_PACKET_CMD_POS` (1u)
- #define `I2C_PACKET_VAL_POS` (2u)
- #define `I2C_PACKET_EOP_POS` (3u)
- #define `I2C_PACKET_SOP` (0xBEu)
- #define `I2C_PACKET_EOP` (0xEFu)
- #define `I2C_STS_CMD_DONE` (0xAAu)
- #define `I2C_STS_CMD_FAIL` (0xEEu)

Funktioner

- void `i2c_init` (void)
- void `i2c_setPacket` (uint8 i2cAddr, uint8 i2cCmd, uint8 i2cVal)
- void `i2c_getPacket` (uint8 i2cAddr, uint8 i2cCmd, uint8 *i2cVal)

4.7.1 Detaljeret beskrivelse

`I2C` modul.

Håndter kommunikation via I2C-busset.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.7.2 #Define-dokumentation

4.7.2.1 #define `I2C_BUFFER_SIZE` (4u)

Defineret på linje 42 i filen `i2c.h`.

Refereret til af `I2C::i2c_rx()` og `I2C::i2c_tx()`.

4.7.2.2 #define `I2C_PACKET_CMD_POS` (1u)

Defineret på linje 47 i filen `i2c.h`.

Refereret til af `I2C::i2c_rx()` og `I2C::i2c_tx()`.

4.7.2.3 #define `I2C_PACKET_EOP` (0xEFu)

Defineret på linje 53 i filen `i2c.h`.

Refereret til af `I2C::i2c_rx()` og `I2C::i2c_tx()`.

4.7.2.4 #define I2C_PACKET_EOP_POS (3u)

Defineret på linje 49 i filen i2c.h.

Refereret til af I2C::i2c_rx() og I2C::i2c_tx().

4.7.2.5 #define I2C_PACKET_SIZE (4u)

Defineret på linje 43 i filen i2c.h.

Refereret til af I2C::i2c_rx() og I2C::i2c_tx().

4.7.2.6 #define I2C_PACKET_SOP (0xBEu)

Defineret på linje 52 i filen i2c.h.

Refereret til af I2C::i2c_rx() og I2C::i2c_tx().

4.7.2.7 #define I2C_PACKET_SOP_POS (0u)

Defineret på linje 46 i filen i2c.h.

Refereret til af I2C::i2c_rx() og I2C::i2c_tx().

4.7.2.8 #define I2C_PACKET_VAL_POS (2u)

Defineret på linje 48 i filen i2c.h.

Refereret til af I2C::i2c_rx() og I2C::i2c_tx().

4.7.2.9 #define I2C_STS_CMD_DONE (0xAAu)

Defineret på linje 56 i filen i2c.h.

Refereret til af I2C::i2c_getPacket(), I2C::i2c_setPacket() og I2C::i2c_tx().

4.7.2.10 #define I2C_STS_CMD_FAIL (0xEEu)

Defineret på linje 57 i filen i2c.h.

Refereret til af I2C::i2c_rx() og I2C::i2c_tx().

4.7.2.11 #define PSoC_Sensor (0x10u)

Defineret på linje 39 i filen i2c.h.

Refereret til af Handler::handler().

4.7.2.12 #define PSoC_XY (0x08u)

Defineret på linje 37 i filen i2c.h.

Refereret til af Handler::handler().

4.7.2.13 `#define PSoC_Z (0x09u)`

Defineret på linje 38 i filen i2c.h.

Refereret til af `Handler::handler()`.

4.7.3 Funktions-dokumentation

4.7.3.1 `void i2c_getPacket (uint8 i2cAddr, uint8 i2cCmd, uint8 * i2cVal)`

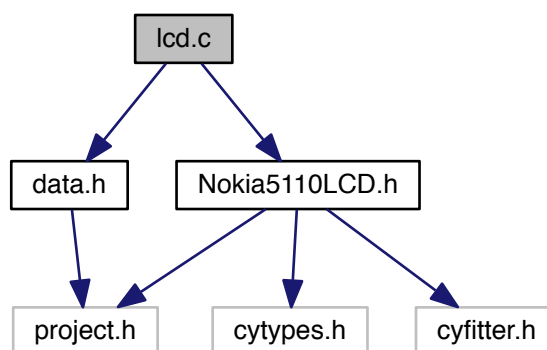
4.7.3.2 `void i2c_init (void)`

4.7.3.3 `void i2c_setPacket (uint8 i2cAddr, uint8 i2cCmd, uint8 i2cVal)`

4.8 lcd.c filreference

LCD modul.

```
#include "data.h"
#include "Nokia5110LCD.h"
Inklusions-afhængighedsgraf for lcd.c:
```



4.8.1 Detaljeret beskrivelse

LCD modul.

Sender tekst til Nokia5110LCD skærmen via dens eksterne kode.

Forfatter

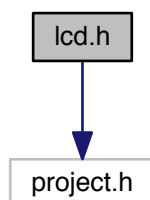
Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.9 lcd.h filreference

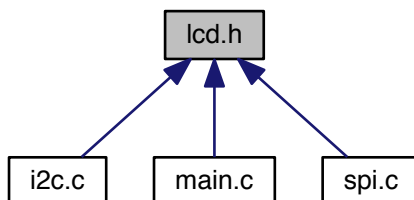
LCD modul.

```
#include <project.h>
```

Inklusions-afhængighedsgraf for lcd.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



Funktioner

- void `lcd_newline` (char *characters)

4.9.1 Detaljeret beskrivelse

LCD modul.

Sender tekst til Nokia5110LCD skærmen via dens eksterne kode.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.9.2 Funktions-dokumentation

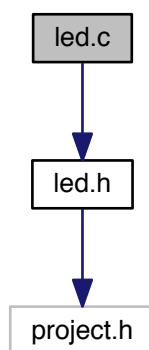
4.9.2.1 void lcd_newline (char * *characters*)

4.10 led.c filreference

LED modul.

```
#include "led.h"
```

Inklusions-afhængighedsgraf for led.c:



4.10.1 Detaljeret beskrivelse

LED modul.

Håndtere PSoC'ens røde, grønne og blå led.

Forfatter

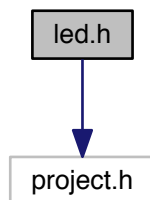
Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.11 led.h filreference

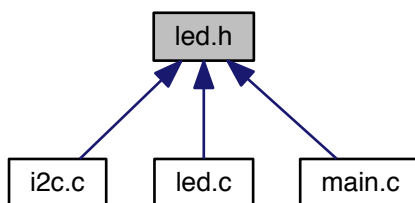
LED modul.


```
#include <project.h>
```

Inklusions-afhængighedsgraf for led.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



#Defines

- #define `LED_ON` (0u)
- #define `LED_OFF` (1u)

Funktioner

- void `setLed` (uint8 red, uint8 green, uint8 blue, uint8 delay)

4.11.1 Detaljeret beskrivelse

`LED` modul.

Håndtere PSoC'ens røde, grønne og blå led.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.11.2 #Define-dokumentation

4.11.2.1 #define LED_OFF (1u)

Defineret på linje 37 i filen led.h.

Refereret til af LED::setLed().

4.11.2.2 #define LED_ON (0u)

Defineret på linje 36 i filen led.h.

Refereret til af LED::setLed().

4.11.3 Funktions-dokumentation

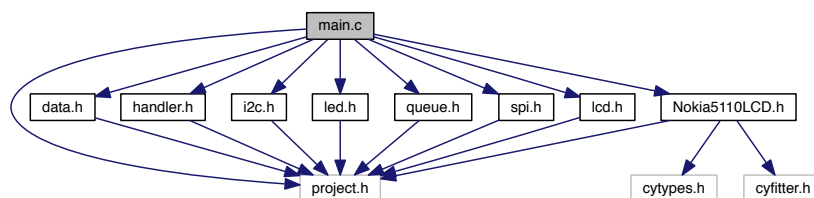
4.11.3.1 void setLed (uint8 red, uint8 green, uint8 blue, uint8 delay)

4.12 main.c filreference

Hovedprogram.

```
#include <project.h>
#include "data.h"
#include "handler.h"
#include "i2c.h"
#include "led.h"
#include "queue.h"
#include "spi.h"
#include "Nokia5110LCD.h"
#include "lcd.h"
```

Inklusions-afhængighedsgraf for main.c:



Funktioner

- int [main](#) ()

4.12.1 Detaljeret beskrivelse

Hovedprogram.

Intilizere modulerne og køre derefter i loop hvor der bliver kontroleret om der er nogle actions i køen der skal håndteres af handleren.

Forfatter

Jeppe Stærk (201271201@uni.au.dk)

4.12.2 Funktions-dokumentation

4.12.2.1 int main ()

Defineret på linje 19 i filen main.c.

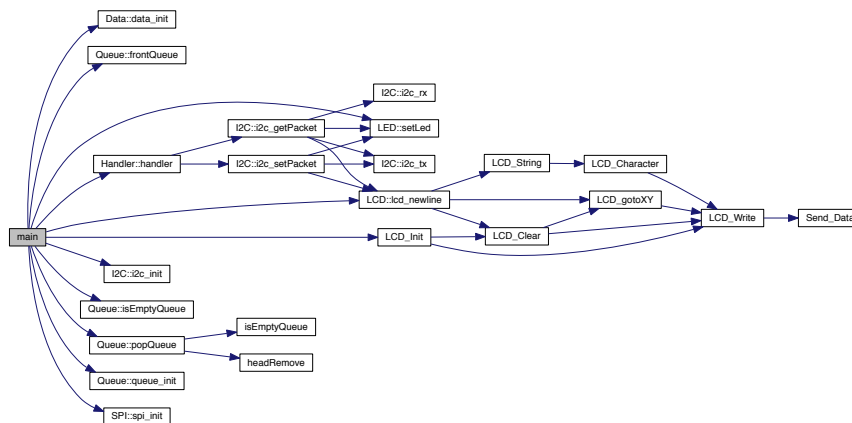
Indeholder referencer til Data::data_init(), Queue::frontQueue(), Handler::handler(), I2C::i2c_init(), Queue::isEmptyQueue(), LCD_Init(), LCD::lcd_newline(), Queue::popQueue(), Queue::queue_init(), LED::setLed() og SPI::spi_init().

```

20 {
21     data_init();
22     queue_init(6u);
23     spi_init();
24     i2c_init();
25     LCD_Init();
26     DEBUG_Start();
27
28     lcd_newline("-----");
29     lcd_newline("< >");
30     lcd_newline("< L.A.M.P! >");
31     lcd_newline("< Prj3Gpr1 >");
32     lcd_newline("< >");
33     lcd_newline("-----");
34
35     setLed(1,0,0,150);
36     setLed(0,1,0,150);
37     setLed(0,0,1,150);
38
39     DEBUG_PutCRLF();
40     DEBUG_PutString("==== Initializing PSoC Master =====");
41     DEBUG_PutCRLF();
42     CyGlobalIntEnable; /* Enable global interrupts. */
43
44     for(;;)
45     {
46         setLed(0,0,0,0);
47
48         while(isEmptyQueue() != 1)
49         {
50             struct Action action;
51             action = frontQueue();
52             if(action.cmd != 0)
53             {
54                 handler(action.cmd, action.val);
55             }
56             popQueue();
57         }
58     }
59 }

```

Her er kald-grafen for denne funktion:

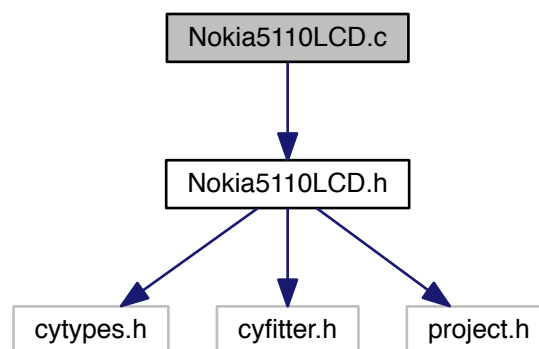


4.13 Nokia5110LCD.c filreference

Nokia5110LCD Modul (Impoteret)

```
#include "Nokia5110LCD.h"
```

Inklusions-afhængighedsgraf for Nokia5110LCD.c:



Funktioner

- void [LCD_Character](#) (uint8 character)
- void [LCD_Clear](#) (void)
- void [LCD_Init](#) (void)
- void [LCD_String](#) (char *characters)
- void [LCD_Write](#) (uint8 data_or_command, uint8 data_value)
- void [Send_Data](#) (int8 value)
- void [LCD_gotoXY](#) (uint8 x, uint8 y)
- void [LCD_Bitmap](#) (char *my_array)

Variable

- static const uint8 `Fonts` `[][FONT_WIDTH]`
- static const char `CypressLogo` `[]`

4.13.1 Detaljeret beskrivelse

Nokia5110LCD Modul (Impoteret)

Impoteret kildekode til Nokia 5110 LSD

Forfatter

Matt (cy.wbz)

Bemærkninger

<https://www.element14.com/community/thread/26122/1/psoc-4-pioneer-kit-community-project>
FullThread=true

4.13.2 Funktions-dokumentation

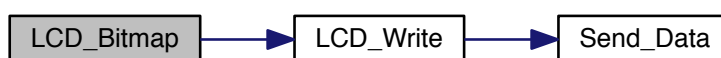
4.13.2.1 void LCD_Bitmap (char * my_array)

Defineret på linje 396 i filen Nokia5110LCD.c.

Indeholder referencer til FONT_HEIGHT, LCD_DATA, LCD_Write(), LCD_X og LCD_Y.

```
397 {  
398     uint16 index4;  
399     for (index4 = 0 ; index4 < (LCD_X * LCD_Y/FONT_HEIGHT) ; index4++)  
400  
401         /* Take one byte at a time and send it to LCD as DATA */  
402         LCD_Write(LCD_DATA, my_array[index4]);  
403 }
```

Her er kald-grafen for denne funktion:



4.13.2.2 void LCD_Character (uint8 character)

Defineret på linje 167 i filen Nokia5110LCD.c.

Indeholder referencer til EMPTY_COLUMN_DATA, FONT_WIDTH, Fonts, LCD_DATA, LCD_Write() og OFFSET_FOR_ASCII.

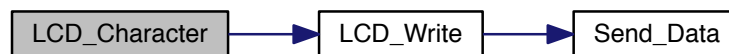
Refereret til af LCD_String().

```

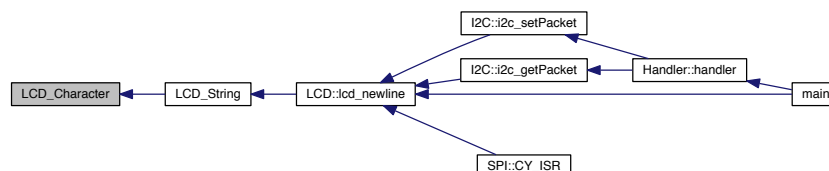
168 {
169     uint16 index1;
170     LCD_Write(LCD_DATA, EMPTY_COLUMN_DATA);
171
172     for (index1 = 0 ; index1 < FONT_WIDTH ; index1++)
173         LCD_Write(LCD_DATA, Fonts[character -
174             OFFSET_FOR_ASCII][index1]);
175     LCD_Write(LCD_DATA, EMPTY_COLUMN_DATA);
176 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.13.2.3 void LCD_Clear (void)

Defineret på linje 193 i filen Nokia5110LCD.c.

Indeholder referencer til COLUMN_BEGINNING, EMPTY_COLUMN_DATA, FONT_HEIGHT, LCD_DATA, LCD_gotoXY(), LCD_Write(), LCD_X, LCD_Y og ROW_BEGINNING.

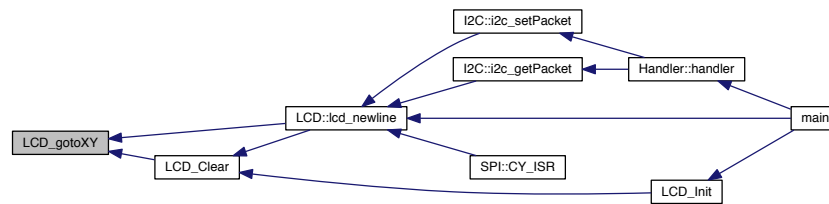
Refereret til af LCD_Init() og LCD::lcd_newline().

```

194 {
195     uint16 index2;
196     LCD_gotoXY(COLUMN_BEGINNING, ROW_BEGINNING);
197
198     for (index2 = 0 ; index2 < (LCD_X * LCD_Y / FONT_HEIGHT) ; index2++)
199         LCD_Write(LCD_DATA, EMPTY_COLUMN_DATA);
200
201     LCD_gotoXY(COLUMN_BEGINNING, ROW_BEGINNING);
202 }

```


Her er kalder-grafen for denne funktion:



4.13.2.5 void LCD_Init (void)

Defineret på linje 220 i filen Nokia5110LCD.c.

Indeholder referencer til BIAS_1_BY_8, BIAS_SYSTEM_MASK, CMD_BIAS_SYSTEM, CMD_DISPLAY_CONTROL, CMD_FUNCTION_SET, CMD_SET_VOP, CMD_TEMP_CONTROL, DELAY_1_MS, DISPLAY_CONTROL_MASK, DISPLAY_NORMAL, FUNCTION_SET_MASK, H_EXTENDED_INST, H_MASK, H_SHIFT, HIGH, LCD_Clear(), LCD_COMMAND, LCD_Write(), LOW, PD_CHIP_ACTIVE, PD_MASK, PD_SHIFT, SET_VOP_5V, SET_VOP_MASK, TEMP_CONTROL_COEFF0, TEMP_CONTROL_MASK, V_HORIZONTAL_ADD, V_MASK og V_SHIFT.

Refereret til af main().

```

221 {
222     /* Enable Gnd, Vcc & Backlight pins */
223     Gnd_Write(LOW);
224     Vcc_Write(HIGH);
225     BL_Write(HIGH);
226
227     /* Reset the LCD - Active Low (1 -> 0 -> 1) */
228     RST_Write(HIGH);
229     CyDelay(DELAY_1_MS);
230     RST_Write(LOW);
231     CyDelay(DELAY_1_MS);
232     RST_Write(HIGH);
233     CyDelay(DELAY_1_MS);
234
235     /* No Power Down, Horizontal Addressing Mode, Extended Instruction set */
236     /* 0x21 = 0010 0001 Instruction : Function Set (H=1)*/
237     LCD_Write(LCD_COMMAND, CMD_FUNCTION_SET | (((
238     PD_CHIP_ACTIVE << PD_SHIFT) & PD_MASK) \
239     | ((V_HORIZONTAL_ADD << V_SHIFT) & V_MASK) | ((
240     H_EXTENDED_INST << H_SHIFT) & H_MASK)) \
241     & FUNCTION_SET_MASK));
242
243     /* Set LCD Vop (Contrast): Try 0xB1(good @ 3.3V) or 0xBF if your display is too dark */
244     /* 0xB1 = 1100 0000 Instruction : Set Vop */
245     LCD_Write(LCD_COMMAND, CMD_SET_VOP | (
246     SET_VOP_5V & SET_VOP_MASK));
247
248     /* Set Temp coefficient */
249     /* 0x04 = 0000 0100 Instruction : Temperature Control */
250     LCD_Write(LCD_COMMAND, CMD_TEMP_CONTROL | (
251     TEMP_CONTROL_COEFF0 & TEMP_CONTROL_MASK));
252
253     /* LCD bias mode 1:48: Try 0x13 or 0x14 */
254     /* 0x13 = 0001 0100 Instruction : Bias System */
255     LCD_Write(LCD_COMMAND, CMD_BIAS_SYSTEM | (
256     BIAS_1_BY_8 & BIAS_SYSTEM_MASK));
257
258     /* We must send 0x20 before modifying the display control mode */
259     /* 0x20 = 0010 0000 Instruction : Function set (H=0)*/
260     LCD_Write(LCD_COMMAND, CMD_FUNCTION_SET | ((
261     PD_CHIP_ACTIVE << (PD_SHIFT - 1)) & FUNCTION_SET_MASK));
262
263     /* Set display control, normal mode. 0x0D for inverse */
264     /* 0x0C = 0000 1101 Instruction : Display control */
265     LCD_Write(LCD_COMMAND, CMD_DISPLAY_CONTROL | (

```

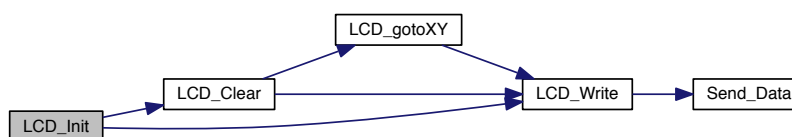


```

        DISPLAY_NORMAL & DISPLAY_CONTROL_MASK));
260
261     /* Clear the LCD screen */
262     LCD_Clear();
263
264     /* Display bitmap image of Cypress Logo on the LCD */
265     // LCD_Bitmap(CypressLogo);
266
267     /* Wait for 1 second before clearing the display */
268     // CyDelay(1000);
269
270     LCD_Clear();
271 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.13.2.6 void LCD_String (char * characters)

Defineret på linje 288 i filen Nokia5110LCD.c.

Indeholder referencer til `LCD_Character()`.

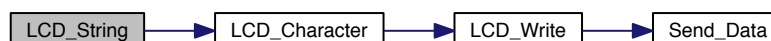
Refereret til af `LCD::lcd_newline()`.

```

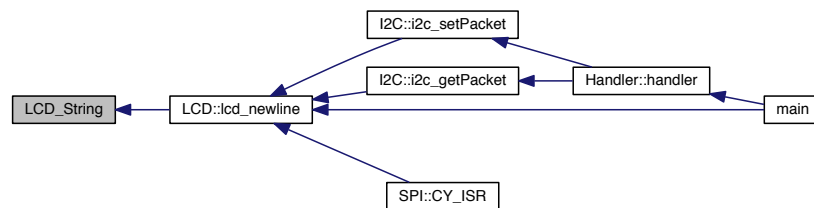
289 {
290     while (*characters)
291         LCD_Character(*characters++);
292 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.13.2.7 void LCD_Write (uint8 data_or_command, uint8 data_value)

Defineret på linje 311 i filen Nokia5110LCD.c.

Indeholder referencer til `DELAY_1_US`, `HIGH`, `LOW` og `Send_Data()`.

Refereret til af `LCD_Bitmap()`, `LCD_Character()`, `LCD_Clear()`, `LCD_gotoXY()` og `LCD_Init()`.

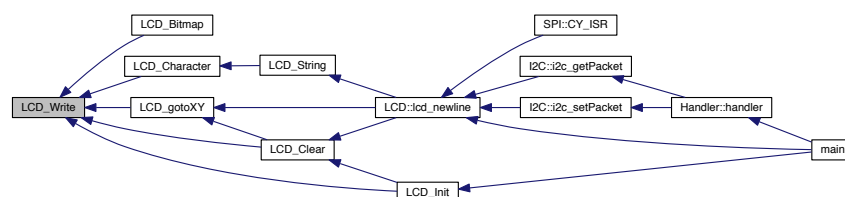
```

312 {
313     CE_Write(LOW);
314     CyDelayUs(DELAY_1_US);
315     DC_Write(data_or_command);
316     Send_Data(data_value);
317     CE_Write(HIGH);
318     CyDelayUs(DELAY_1_US);
319 }
  
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:

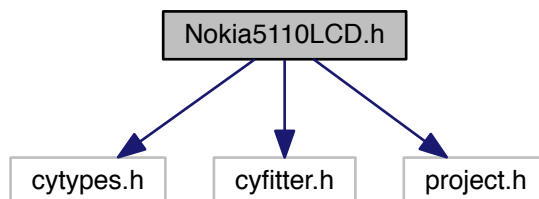


4.14 Nokia5110LCD.h filreference

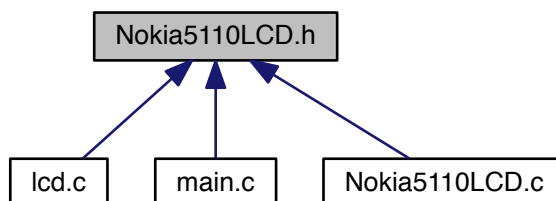
Nokia5110LCD Modul (Impoteret)

```
#include "cytypes.h"  
#include "cyfitter.h"  
#include "project.h"
```

Inklusions-afhængighedsgraf for Nokia5110LCD.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



#Defines

- #define HIGH 1
- #define LOW 0
- #define LCD_COMMAND LOW
- #define LCD_DATA HIGH
- #define LCD_X 84
- #define LCD_Y 48
- #define FONT_HEIGHT 8
- #define FONT_WIDTH 5
- #define EMPTY_COLUMN_DATA 0x00u
- #define OFFSET_FOR_ASCII 0x20u
- #define MSb_POSITION 0x80u
- #define DELAY_1_US 0x01u

- #define `DELAY_1_MS` 0x01u
- #define `ROW_BEGINNING` 0x00u
- #define `COLUMN_BEGINNING` 0x00u
- #define `SHIFT_LEFT_BY_1` 0x01u
- #define `CMD_NOP` 0x00u
- #define `CMD_FUNCTION_SET` 0x20u
- #define `FUNCTION_SET_MASK` 0x07u
- #define `PD_SHIFT` 0x02u
- #define `PD_MASK` (0x01u << PD_SHIFT)
- #define `PD_CHIP_ACTIVE` 0x00u
- #define `PD_CHIP_POWER_DOWN` 0x01u
- #define `V_SHIFT` 0x01u
- #define `V_MASK` (0x01u << V_SHIFT)
- #define `V_HORIZONTAL_ADD` 0x00u
- #define `V_VERTICAL_ADD` 0x01u
- #define `H_SHIFT` 0x00u
- #define `H_MASK` (0x01u << H_SHIFT)
- #define `H_BASIC_INST` 0x00u
- #define `H_EXTENDED_INST` 0x01u
- #define `CMD_DISPLAY_CONTROL` 0x08u
- #define `DISPLAY_CONTROL_MASK` 0x05u
- #define `DISPLAY_BLANK` 0x00u
- #define `DISPLAY_NORMAL` 0x04u
- #define `DISPLAY_ALL_SEG_ON` 0x01u
- #define `DISPLAY_INVERSE` 0x05u
- #define `CMD_COLUMN_DATA` 0x80u
- #define `COLUMN_DATA_MASK` 0x7Fu
- #define `CMD_ROW_DATA` 0x40u
- #define `ROW_DATA_MASK` 0x07u
- #define `CMD_TEMP_CONTROL` 0x04u
- #define `TEMP_CONTROL_MASK` 0x03u
- #define `TEMP_CONTROL_COEFF0` 0x00u
- #define `TEMP_CONTROL_COEFF1` 0x01u
- #define `TEMP_CONTROL_COEFF2` 0x02u
- #define `TEMP_CONTROL_COEFF3` 0x03u
- #define `CMD_BIAS_SYSTEM` 0x10u
- #define `BIAS_SYSTEM_MASK` 0x07u
- #define `BIAS_1_BY_11` 0x00u
- #define `BIAS_1_BY_10` 0x01u
- #define `BIAS_1_BY_9` 0x02u
- #define `BIAS_1_BY_8` 0x03u
- #define `BIAS_1_BY_7` 0x04u
- #define `BIAS_1_BY_6` 0x05u
- #define `BIAS_1_BY_5` 0x06u
- #define `BIAS_1_BY_4` 0x07u
- #define `CMD_SET_VOP` 0x80u
- #define `SET_VOP_MASK` 0xFFu
- #define `SET_VOP_5V` 0x40u
- #define `SET_VOP_3V` 0x31u

Funktioner

- void [LCD_Character](#) (uint8)
- void [LCD_Clear](#) (void)
- void [LCD_Init](#) (void)
- void [LCD_String](#) (char *)
- void [LCD_Write](#) (uint8, uint8)
- void [Send_Data](#) (int8)
- void [LCD_gotoXY](#) (uint8, uint8)
- void [LCD_Bitmap](#) (char *)

4.14.1 Detaljeret beskrivelse

Nokia5110LCD Modul (Impoteret)

Impoteret kildekode til Nokia 5110 LSD

Forfatter

Matt (cy.wbz)

Bemærkninger

<https://www.element14.com/community/thread/26122/1/psoc-4-pioneer-kit-community-project/FullThread=true>

4.14.2 #Define-dokumentation

4.14.2.1 #define BIAS_1_BY_10 0x01u

Defineret på linje 116 i filen Nokia5110LCD.h.

4.14.2.2 #define BIAS_1_BY_11 0x00u

Defineret på linje 115 i filen Nokia5110LCD.h.

4.14.2.3 #define BIAS_1_BY_4 0x07u

Defineret på linje 122 i filen Nokia5110LCD.h.

4.14.2.4 #define BIAS_1_BY_5 0x06u

Defineret på linje 121 i filen Nokia5110LCD.h.

4.14.2.5 #define BIAS_1_BY_6 0x05u

Defineret på linje 120 i filen Nokia5110LCD.h.

4.14.2.6 #define BIAS_1_BY_7 0x04u

Defineret på linje 119 i filen Nokia5110LCD.h.

4.14.2.7 #define BIAS_1_BY_8 0x03u

Defineret på linje 118 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.8 #define BIAS_1_BY_9 0x02u

Defineret på linje 117 i filen Nokia5110LCD.h.

4.14.2.9 #define BIAS_SYSTEM_MASK 0x07u

Defineret på linje 113 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.10 #define CMD_BIAS_SYSTEM 0x10u

Defineret på linje 112 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.11 #define CMD_COLUMN_DATA 0x80u

Defineret på linje 88 i filen Nokia5110LCD.h.

Refereret til af LCD_gotoXY().

4.14.2.12 #define CMD_DISPLAY_CONTROL 0x08u

Defineret på linje 77 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.13 #define CMD_FUNCTION_SET 0x20u

Defineret på linje 53 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.14 #define CMD_NOP 0x00u

Defineret på linje 48 i filen Nokia5110LCD.h.

4.14.2.15 #define CMD_ROW_DATA 0x40u

Defineret på linje 94 i filen Nokia5110LCD.h.

Refereret til af LCD_gotoXY().

4.14.2.16 #define CMD_SET_VOP 0x80u

Defineret på linje 129 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.17 #define CMD_TEMP_CONTROL 0x04u

Defineret på linje 100 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.18 #define COLUMN_BEGINNING 0x00u

Defineret på linje 42 i filen Nokia5110LCD.h.

Refereret til af LCD_Clear().

4.14.2.19 #define COLUMN_DATA_MASK 0x7Fu

Defineret på linje 89 i filen Nokia5110LCD.h.

Refereret til af LCD_gotoXY().

4.14.2.20 #define DELAY_1_MS 0x01u

Defineret på linje 40 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.21 #define DELAY_1_US 0x01u

Defineret på linje 39 i filen Nokia5110LCD.h.

Refereret til af LCD_Write() og Send_Data().

4.14.2.22 #define DISPLAY_ALL_SEG_ON 0x01u

Defineret på linje 82 i filen Nokia5110LCD.h.

4.14.2.23 #define DISPLAY_BLANK 0x00u

Defineret på linje 80 i filen Nokia5110LCD.h.

4.14.2.24 #define DISPLAY_CONTROL_MASK 0x05u

Defineret på linje 78 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.25 #define DISPLAY_INVERSE 0x05u

Defineret på linje 83 i filen Nokia5110LCD.h.

4.14.2.26 #define DISPLAY_NORMAL 0x04u

Defineret på linje 81 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.27 #define EMPTY_COLUMN_DATA 0x00u

Defineret på linje 36 i filen Nokia5110LCD.h.

Refereret til af LCD_Character() og LCD_Clear().

4.14.2.28 #define FONT_HEIGHT 8

Defineret på linje 33 i filen Nokia5110LCD.h.

Refereret til af LCD_Bitmap(), LCD_Clear() og Send_Data().

4.14.2.29 #define FONT_WIDTH 5

Defineret på linje 34 i filen Nokia5110LCD.h.

Refereret til af LCD_Character().

4.14.2.30 #define FUNCTION_SET_MASK 0x07u

Defineret på linje 54 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.31 #define H_BASIC_INST 0x00u

Defineret på linje 71 i filen Nokia5110LCD.h.

4.14.2.32 #define H_EXTENDED_INST 0x01u

Defineret på linje 72 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.33 #define H_MASK (0x01u << H_SHIFT)

Defineret på linje 70 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.34 #define H_SHIFT 0x00u

Defineret på linje 69 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.35 #define HIGH 1

Defineret på linje 24 i filen Nokia5110LCD.h.

Refereret til af LCD_Init(), LCD_Write() og Send_Data().

4.14.2.36 #define LCD_COMMAND LOW

Defineret på linje 27 i filen Nokia5110LCD.h.

Refereret til af LCD_gotoXY() og LCD_Init().

4.14.2.37 #define LCD_DATA HIGH

Defineret på linje 28 i filen Nokia5110LCD.h.

Refereret til af LCD_Bitmap(), LCD_Character() og LCD_Clear().

4.14.2.38 #define LCD_X 84

Defineret på linje 30 i filen Nokia5110LCD.h.

Refereret til af LCD_Bitmap() og LCD_Clear().

4.14.2.39 #define LCD_Y 48

Defineret på linje 31 i filen Nokia5110LCD.h.

Refereret til af LCD_Bitmap() og LCD_Clear().

4.14.2.40 #define LOW 0

Defineret på linje 25 i filen Nokia5110LCD.h.

Refereret til af LCD_Init(), LCD_Write() og Send_Data().

4.14.2.41 #define MSb_POSITION 0x80u

Defineret på linje 38 i filen Nokia5110LCD.h.

Refereret til af Send_Data().

4.14.2.42 #define OFFSET_FOR_ASCII 0x20u

Defineret på linje 37 i filen Nokia5110LCD.h.

Refereret til af LCD_Character().

4.14.2.43 #define PD_CHIP_ACTIVE 0x00u

Defineret på linje 59 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.44 #define PD_CHIP_POWER_DOWN 0x01u

Defineret på linje 60 i filen Nokia5110LCD.h.

4.14.2.45 #define PD_MASK (0x01u << PD_SHIFT)

Defineret på linje 58 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.46 #define PD_SHIFT 0x02u

Defineret på linje 57 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.47 #define ROW_BEGINNING 0x00u

Defineret på linje 41 i filen Nokia5110LCD.h.

Refereret til af LCD_Clear().

4.14.2.48 #define ROW_DATA_MASK 0x07u

Defineret på linje 95 i filen Nokia5110LCD.h.

Refereret til af LCD_gotoXY().

4.14.2.49 #define SET_VOP_3V 0x31u

Defineret på linje 133 i filen Nokia5110LCD.h.

4.14.2.50 #define SET_VOP_5V 0x40u

Defineret på linje 132 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.51 #define SET_VOP_MASK 0xFFu

Defineret på linje 130 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.52 #define SHIFT_LEFT_BY_1 0x01u

Defineret på linje 43 i filen Nokia5110LCD.h.

Refereret til af Send_Data().

4.14.2.53 #define TEMP_CONTROL_COEFF0 0x00u

Defineret på linje 104 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.54 #define TEMP_CONTROL_COEFF1 0x01u

Defineret på linje 105 i filen Nokia5110LCD.h.

4.14.2.55 #define TEMP_CONTROL_COEFF2 0x02u

Defineret på linje 106 i filen Nokia5110LCD.h.

4.14.2.56 #define TEMP_CONTROL_COEFF3 0x03u

Defineret på linje 107 i filen Nokia5110LCD.h.

4.14.2.57 #define TEMP_CONTROL_MASK 0x03u

Defineret på linje 101 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.58 #define V_HORIZONTAL_ADD 0x00u

Defineret på linje 65 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.59 #define V_MASK (0x01u << V_SHIFT)

Defineret på linje 64 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.60 #define V_SHIFT 0x01u

Defineret på linje 63 i filen Nokia5110LCD.h.

Refereret til af LCD_Init().

4.14.2.61 #define V_VERTICAL_ADD 0x01u

Defineret på linje 66 i filen Nokia5110LCD.h.

4.14.3 Funktions-dokumentation

4.14.3.1 void LCD_Bitmap (char *)

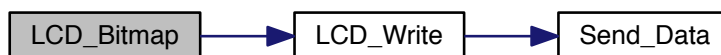
Defineret på linje 396 i filen Nokia5110LCD.c.

Indeholder referencer til FONT_HEIGHT, LCD_DATA, LCD_Write(), LCD_X og LCD_Y.

```

397 {
398     uint16 index4;
399     for (index4 = 0 ; index4 < (LCD_X * LCD_Y/FONT_HEIGHT) ; index4++)
400
401         /* Take one byte at a time and send it to LCD as DATA */
402         LCD_Write(LCD_DATA, my_array[index4]);
403 }
```

Her er kald-grafen for denne funktion:



4.14.3.2 void LCD_Character (uint8)

Defineret på linje 167 i filen Nokia5110LCD.c.

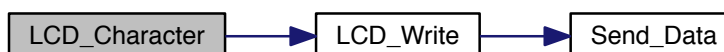
Indeholder referencer til EMPTY_COLUMN_DATA, FONT_WIDTH, Fonts, LCD_DATA, LCD_Write() og OFFSET_FOR_ASCII.

Refereret til af LCD_String().

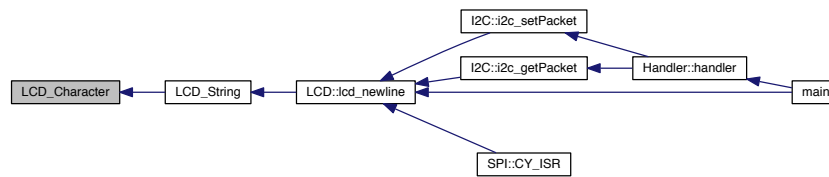
```

168 {
169     uint16 index1;
170     LCD_Write(LCD_DATA, EMPTY_COLUMN_DATA);
171
172     for (index1 = 0 ; index1 < FONT_WIDTH ; index1++)
173         LCD_Write(LCD_DATA, Fonts[character -
174             OFFSET_FOR_ASCII][index1]);
175     LCD_Write(LCD_DATA, EMPTY_COLUMN_DATA);
176 }
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.14.3.3 void LCD_Clear (void)

Defineret på linje 193 i filen Nokia5110LCD.c.

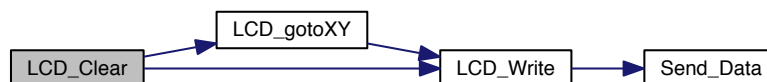
Indeholder referencer til COLUMN_BEGINNING, EMPTY_COLUMN_DATA, FONT_HEIGHT, LCD_DATA, LCD_gotoXY(), LCD_Write(), LCD_X, LCD_Y og ROW_BEGINNING.

Refereret til af LCD_Init() og LCD::lcd_newline().

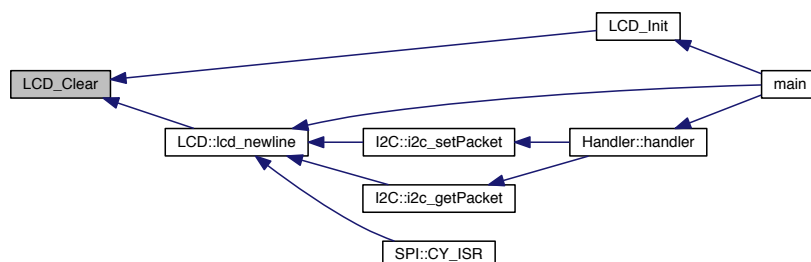
```

194 {
195     uint16 index2;
196     LCD_gotoXY(COLUMN_BEGINNING, ROW_BEGINNING);
197
198     for (index2 = 0 ; index2 < (LCD_X * LCD_Y / FONT_HEIGHT) ; index2++)
199         LCD_Write(LCD_DATA, EMPTY_COLUMN_DATA);
200
201     LCD_gotoXY(COLUMN_BEGINNING, ROW_BEGINNING);
202 }
  
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.14.3.4 void LCD_gotoXY (uint8 , uint8)

Defineret på linje 375 i filen Nokia5110LCD.c.

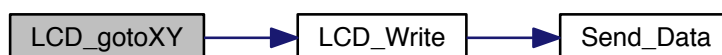
Indeholder referencer til CMD_COLUMN_DATA, CMD_ROW_DATA, COLUMN_DATA_MASK, LCD_COMMAND, LCD_Write() og ROW_DATA_MASK.

Refereret til af LCD_Clear() og LCD::lcd_newline().

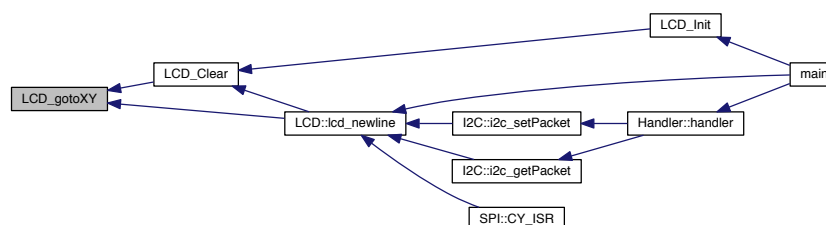
```

376 {
377     LCD_Write(LCD_COMMAND, CMD_COLUMN_DATA | (x &
COLUMN_DATA_MASK)); // Column
378     LCD_Write(LCD_COMMAND, CMD_ROW_DATA | (y &
ROW_DATA_MASK)); // Row : Last 3 bits are valid
379 }
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.14.3.5 void LCD_Init (void)

Defineret på linje 220 i filen Nokia5110LCD.c.

Indeholder referencer til BIAS_1_BY_8, BIAS_SYSTEM_MASK, CMD_BIAS_SYSTEM, CMD_DISPLAY_CONTROL, CMD_FUNCTION_SET, CMD_SET_VOP, CMD_TEMP_CONTROL, DELAY_1_MS, DISPLAY_CONTROL_MASK, DISPLAY_NORMAL, FUNCTION_SET_MASK, H_EXTENDED_INST, H_MASK, H_SHIFT, HIGH, LCD_Clear(), LCD_COMMAND, LCD_Write(), LOW, PD_CHIP_ACTIVE, PD_MASK, PD_SHIFT, SET_VOP_5V, SET_VOP_MASK, TEMP_CONTROL_COEFF0, TEMP_CONTROL_MASK, V_HORIZONTAL_ADD, V_MASK og V_SHIFT.

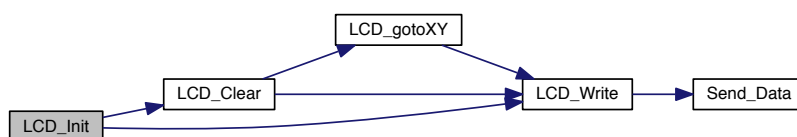
Refereret til af main().

```

221 {
222     /* Enable Gnd, Vcc & Backlight pins */
223     Gnd_Write(LOW);
224     Vcc_Write(HIGH);
225     BL_Write(HIGH);
226
227     /* Reset the LCD - Active Low (1 -> 0 -> 1) */
228     RST_Write(HIGH);
229     CyDelay(DELAY_1_MS);
230     RST_Write(LOW);
231     CyDelay(DELAY_1_MS);
232     RST_Write(HIGH);
233     CyDelay(DELAY_1_MS);
234
235     /* No Power Down, Horizontal Addressing Mode, Extended Instruction set */
236     /* 0x21 = 0010 0001 Instruction : Function Set (H=1)*/
237     LCD_Write(LCD_COMMAND, CMD_FUNCTION_SET | (((
PD_CHIP_ACTIVE << PD_SHIFT) & PD_MASK) \
238 | ((V_HORIZONTAL_ADD << V_SHIFT) & V_MASK) | ((
H_EXTENDE INST << H_SHIFT) & H_MASK)) \
239 & FUNCTION_SET_MASK));
240
241     /* Set LCD Vop (Contrast): Try 0xB1(good @ 3.3V) or 0xBF if your display is too dark */
242     /* 0xB1 = 1100 0000 Instruction : Set Vop */
243     LCD_Write(LCD_COMMAND, CMD_SET_VOP | (
SET_VOP_5V & SET_VOP_MASK));
244
245     /* Set Temp coefficient */
246     /* 0x04 = 0000 0100 Instruction : Temperature Control */
247     LCD_Write(LCD_COMMAND, CMD_TEMP_CONTROL | (
TEMP_CONTROL_COEFF0 & TEMP_CONTROL_MASK));
248
249     /* LCD bias mode 1:48: Try 0x13 or 0x14 */
250     /* 0x13 = 0001 0100 Instruction : Bias System */
251     LCD_Write(LCD_COMMAND, CMD_BIAS_SYSTEM | (
BIAS_1_BY_8 & BIAS_SYSTEM_MASK));
252
253     /* We must send 0x20 before modifying the display control mode */
254     /* 0x20 = 0010 0000 Instruction : Function set (H=0)*/
255     LCD_Write(LCD_COMMAND, CMD_FUNCTION_SET | ((
PD_CHIP_ACTIVE << (PD_SHIFT - 1)) & FUNCTION_SET_MASK));
256
257     /* Set display control, normal mode. 0x0D for inverse */
258     /* 0x0C = 0000 1101 Instruction : Display control */
259     LCD_Write(LCD_COMMAND, CMD_DISPLAY_CONTROL | (
DISPLAY_NORMAL & DISPLAY_CONTROL_MASK));
260
261     /* Clear the LCD screen */
262     LCD_Clear();
263
264     /* Display bitmap image of Cypress Logo on the LCD */
265     // LCD_Bitmap(CypressLogo);
266
267     /* Wait for 1 second before clearing the display */
268     // CyDelay(1000);
269
270     LCD_Clear();
271 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.14.3.6 void LCD_String (char *)

Defineret på linje 288 i filen Nokia5110LCD.c.

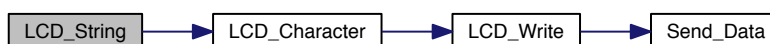
Indeholder referencer til `LCD_Character()`.

Refereret til af `LCD::lcd_newline()`.

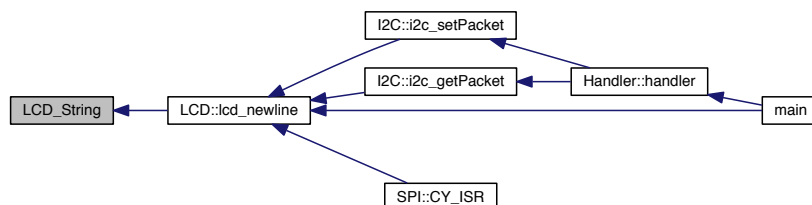
```

289 {
290     while (*characters)
291         LCD_Character(*characters++);
292 }
  
```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.14.3.7 void LCD_Write (uint8 , uint8)

Defineret på linje 311 i filen Nokia5110LCD.c.

Indeholder referencer til DELAY_1_US, HIGH, LOW og Send_Data().

Refereret til af LCD_Bitmap(), LCD_Character(), LCD_Clear(), LCD_gotoXY() og LCD_Init().

```

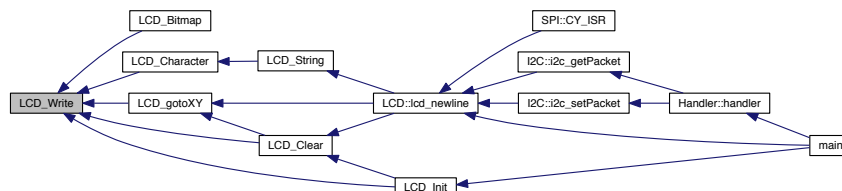
312 {
313     CE_Write(LOW);
314     CyDelayUs(DELAY_1_US);
315     DC_Write(data_or_command);
316     Send_Data(data_value);
317     CE_Write(HIGH);
318     CyDelayUs(DELAY_1_US);
319 }

```

Her er kald-grafen for denne funktion:



Her er kalder-grafen for denne funktion:



4.14.3.8 void Send_Data (int8)

Defineret på linje 336 i filen Nokia5110LCD.c.

Indeholder referencer til DELAY_1_US, FONT_HEIGHT, HIGH, LOW, MSb_POSITION og SHIFT_LEFT_BY_1.

Refereret til af LCD_Write().

```

337 {
338     uint8 index3;
339     for (index3 = 0; index3 < FONT_HEIGHT; index3++)
340     {
341         /* Take one bit (MSb) at a time and send it to Data Input pin of LCD */
342         if(MSb_POSITION == (value & MSb_POSITION))
343             Din_Write(HIGH);
344         else
345             Din_Write(LOW);
346     }
347 }

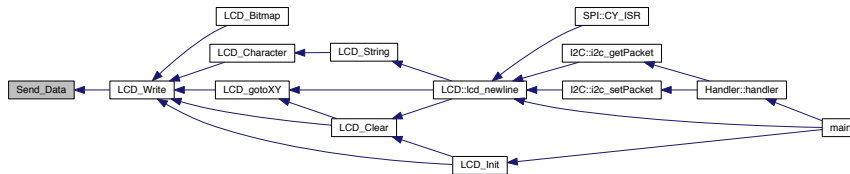
```

```

346
347      /* After setting the Data value on Din pin, toggle the Clock so that LCD can read Din */
348      Clk_Write(HIGH);
349      CyDelayUs(DELAY_1_US);
350      Clk_Write(LOW);
351      CyDelayUs(DELAY_1_US);
352
353      /* Left shift the value before processing next bit */
354      value <<= SHIFT_LEFT_BY_1;
355  }
356 }

```

Her er kalder-grafen for denne funktion:



4.15 queue.c filreference

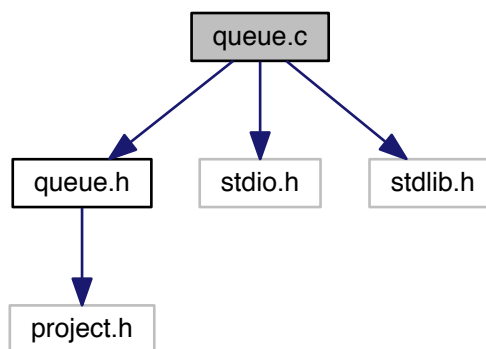
Queue modul.

```

#include "queue.h"
#include <stdio.h>
#include <stdlib.h>

```

Inklusions-afhængighedsgraf for queue.c:



Datastrukturer

- struct [Node](#)

[Node struct. Mere...](#)

Funktioner

- static void `headInsert` (struct `Node` **headPtr, const struct `Action` data)
- static void `headRemove` (struct `Node` **headPtr)
- static void `backInsert` (struct `Node` **backPtr, const struct `Action` data)

4.15.1 Detaljeret beskrivelse

`Queue` modul.

En FIFO kø der er opbygget af en single linked liste.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.15.2 Datastruktur-documentation

4.15.2.1 struct Node

`Node` struct.

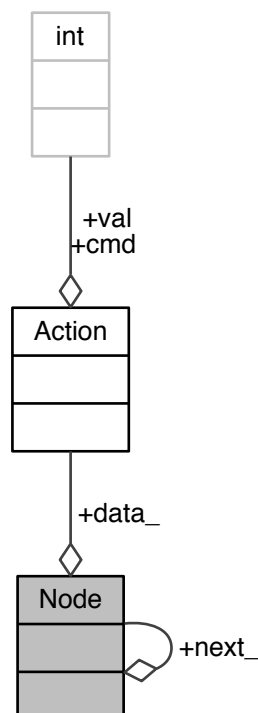
En struct til at oprette et element der kan indsættes i køen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 20 i filen queue.c.

Samarbejdsdiagram for Node:



Data-felter

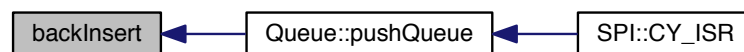
struct Action	data↔ —	Data til køen
struct Node *	next↔ —	Pointer til næste node i køen

4.15.3 Funktions-dokumentation

4.15.3.1 static void backInsert (struct Node ** backPtr, const struct Action data) [static]

Refereret til af Queue::pushQueue().

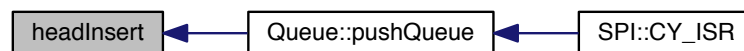
Her er kalder-grafen for denne funktion:



4.15.3.2 static void headInsert (struct Node ** headPtr, const struct Action data) [static]

Refereret til af Queue::pushQueue().

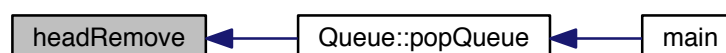
Her er kalder-grafen for denne funktion:



4.15.3.3 static void headRemove (struct Node ** headPtr) [static]

Refereret til af Queue::popQueue().

Her er kalder-grafen for denne funktion:

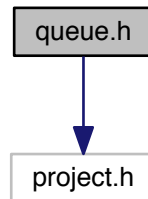


4.16 queue.h filreference

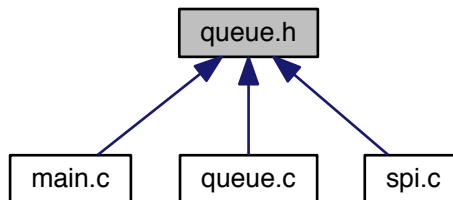
[Queue](#) modul.

```
#include <project.h>
```

Inklusions-afhængighedsgraf for queue.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



Datastrukturer

- struct [Action](#)
[Action struct. Mere...](#)

Funktioner

- void [queue_init](#) (uint8 queueMaxSize)
- void [pushQueue](#) (const struct [Action](#) data)
- void [popQueue](#) (void)
- struct [Action](#) [frontQueue](#) (void)
- uint8 [isEmptyQueue](#) (void)

4.16.1 Detaljeret beskrivelse

[Queue](#) modul.

En FIFO kø der er opbygget af en single linked liste.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.16.2 Datastruktur-documentation

4.16.2.1 struct Action

[Action](#) struct.

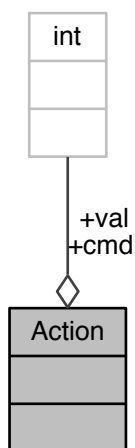
Structen kan indeholde en kommando og tilhørende værdi, som kan indsættes i FIFO køen.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

Defineret på linje 33 i filen queue.h.

Samarbejdsdiagram for Action:



Data-felter

int	cmd	Kommando
int	val	Værdi

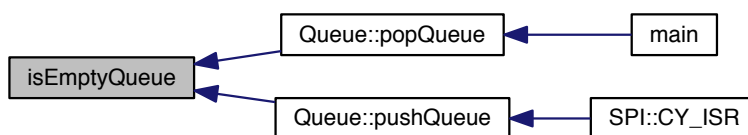
4.16.3 Funktions-dokumentation

4.16.3.1 struct Action frontQueue (void)

4.16.3.2 uint8 isEmptyQueue (void)

Refereret til af Queue::popQueue() og Queue::pushQueue().

Her er kalder-grafen for denne funktion:



4.16.3.3 void popQueue (void)

4.16.3.4 void pushQueue (const struct Action data)

4.16.3.5 void queue_init (uint8 queueMaxSize)

4.17 spi.c filreference

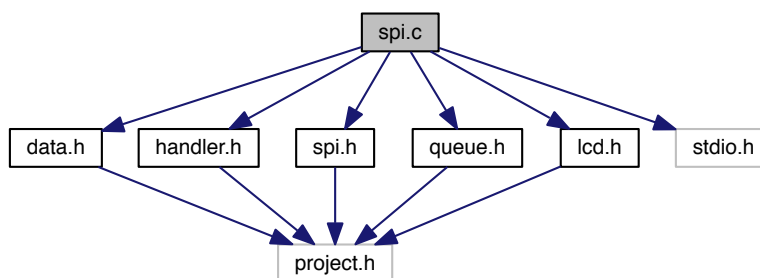
SPI modul.

```

#include "data.h"
#include "handler.h"
#include "spi.h"
#include "queue.h"
#include "lcd.h"
#include <stdio.h>

```

Inklusions-afhængighedsgraf for spi.c:



4.17.1 Detaljeret beskrivelse

SPI modul.

Håndter kommunikation via SPI-busset

Forfatter

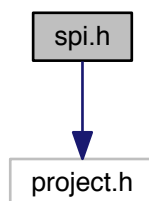
Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.18 spi.h filreference

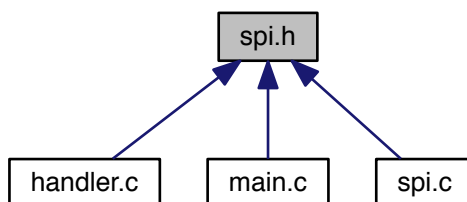
SPI modul.

```
#include <project.h>
```

Inklusions-afhængighedsgraf for spi.h:



Denne graf viser, hvilke filer der direkte eller indirekte inkluderer denne fil:



#Defines

- #define SPI_BUFFER_SIZE (1u)
- #define SPI_PACKET_SIZE (1u)
- #define SPI_PACKET_DATA_POS (0u)
- #define SPI_STS_CMD_DONE (0xAAAAu)
- #define SPI_STS_CMD_FAIL (0xEEEEu)

Funktioner

- void [spi_init](#) (void)
- [CY_ISR_PROTO](#) (isr_spi_rx)

4.18.1 Detaljeret beskrivelse

[SPI](#) modul.

Håndter kommunikation via SPI-busset.

Forfatter

Jeppe Stærk Antonsen (201271201@uni.au.dk)

4.18.2 #Define-dokumentation

4.18.2.1 #define SPI_BUFFER_SIZE (1u)

Defineret på linje 37 i filen spi.h.

4.18.2.2 #define SPI_PACKET_DATA_POS (0u)

Defineret på linje 41 i filen spi.h.

Refereret til af SPI::CY_ISR().

4.18.2.3 #define SPI_PACKET_SIZE (1u)

Defineret på linje 38 i filen spi.h.

Refereret til af SPI::CY_ISR().

4.18.2.4 #define SPI_STS_CMD_DONE (0xAAAAu)

Defineret på linje 44 i filen spi.h.

4.18.2.5 #define SPI_STS_CMD_FAIL (0xEEEEu)

Defineret på linje 45 i filen spi.h.

4.18.3 Funktions-dokumentation

4.18.3.1 CY_ISR_PROTO (isr_spi_rx)

4.18.3.2 void spi_init (void)

