```c
 #include "stm32f10x.h"
 #include "project.h"
 #include <string.h>


void lcd_IO_init (void)
{
    //Enable peripheral clocks for various ports and subsystems
    //Bit 4: Port C Bit3: Port B Bit 2: Port A
    RCC->APB2ENR |=  RCC_APB2ENR_IOPCEN | RCC_APB2ENR_IOPBEN
        | RCC_APB2ENR_IOPAEN ;

    //Set the config and mode bits for Port C bits 7 to 0 so they will
    // be push-pull outputs (up to 50 MHz)
    GPIOC->CRL |= GPIO_CRL_MODE7 | GPIO_CRL_MODE6 | GPIO_CRL_MODE5 | GPIO_CRL_MODE4 | GPIO_CRL_MODE3 |
GPIO_CRL_MODE2 | GPIO_CRL_MODE1 | GPIO_CRL_MODE0 ;
    GPIOC->CRL &= ~GPIO_CRL_CNF7 & ~GPIO_CRL_CNF6 & ~GPIO_CRL_CNF5 & ~GPIO_CRL_CNF4 & ~GPIO_CRL_CNF3 &
~GPIO_CRL_CNF2 & ~GPIO_CRL_CNF1 & ~GPIO_CRL_CNF0 ;
    //Set the config and mode bits for Port B bits 0, 1, and 5 so they will
    // be push-pull outputs (up to 50 MHz)
    GPIOB->CRL |= GPIO_CRL_MODE5 | GPIO_CRL_MODE1 | GPIO_CRL_MODE0;
    GPIOB->CRH |= GPIO_CRH_MODE10;
    GPIOB->CRL &= ~GPIO_CRL_CNF5 & ~GPIO_CRL_CNF1 & ~GPIO_CRL_CNF0;
    GPIOB->CRH &= ~GPIO_CRH_CNF10;

    delay(90000);
    commandToLCD(0x38);
    commandToLCD(0x38);
    commandToLCD(0x38);
    commandToLCD(0x38);
    commandToLCD(0x0F);
    commandToLCD(0x01);
    commandToLCD(0x06);
}
/*
* Name: commandToLCD
* Type: PUBLIC
* Parameters: a single byte of command information for the LCD controller
* Returns: nothing
* Description: This function generates control timing and data signals to send one command byte to the
LCD
*/
void commandToLCD(uint8_t data)
{
GPIOB->BSRR = LCD_CM_ENA; //RS low, E high
// GPIOC->ODR = data; //BAD: may affect upper bits on port C
GPIOC->ODR &= 0xFF00; //GOOD: clears the low bits without affecting high bits
 GPIOC->ODR |= data; //GOOD: only affects lowest 8 bits of Port C
delay(8000);
 GPIOB->BSRR = LCD_CM_DIS; //RS low, E low
 delay(80000);
}

void dataToLCD(uint8_t data)
{
GPIOB->BSRR = LCD_DM_ENA; //RS low, E high
// GPIOC->ODR = data; //BAD: may affect upper bits on port C
GPIOC->ODR &= 0xFF00; //GOOD: clears the low bits without affecting high bits
 GPIOC->ODR |= data; //GOOD: only affects lowest 8 bits of Port C
delay(8000);
 GPIOB->BSRR = LCD_DM_DIS; //RS low, E low
 delay(80000);
}

void stringToLCD(char * message)
{
  int i=0;
  uint16_t messageLength = strlen(message);
  for (i=0; i<messageLength; ++i)
  {
    dataToLCD(*message);
    ++message;
  }
}

uint16_t Hex2Ascii(uint8_t hexval)
  {
    uint8_t ascval;
    if(hexval < 0xA)
      ascval = hexval + 0x30;
    else
      ascval = hexval + 0x37;
    return(ascval);
  }

  void printToLCD1(uint16_t readVal)
{
  int i=0;
  int shiftAmount = 28;
  uint16_t tempVal;

  commandToLCD(LCD_CLR);
  stringToLCD("Stall:1");
  commandToLCD(LCD_LN2);
  stringToLCD("Temp:0x");

  for (i=0; i<8; ++i)
  {
    tempVal = Hex2Ascii((readVal >> shiftAmount) & 0xF);

    shiftAmount -= 4;

    dataToLCD(tempVal);
  }
}
  void printToLCD2(uint16_t readVal)
{
  int i=0;
  int shiftAmount = 28;
  uint16_t tempVal;

  commandToLCD(LCD_CLR);
  stringToLCD("Stall:2");
  commandToLCD(LCD_LN2);
  stringToLCD("Temp:0x");

  for (i=0; i<8; ++i)
  {
    tempVal = Hex2Ascii((readVal >> shiftAmount) & 0xF);

    shiftAmount -= 4;

    dataToLCD(tempVal);
  }
}
```