```
/*-------------------------------------------------------------------------
 * Name:    DDS.c
 * Purpose: Fucntions to initialise the DDS, set default data, and accept new
            frequencies from the user.
 * Note(s):
 *-------------------------------------------------------------------------*/

#include "STM32F4xx.h"
#include "DDS.h"
#include "main_2.h"
#include <math.h>

#define DDS_CLOCK 125000000
#define CLOCK 4 /* W_CLK pin */
#define LOAD   5 /* FQ_UP pin*/
#define DATA   3 /* DATA pin */

/*-------------------------------------------------------------------------
  initialize DDS for serial communication
 *-------------------------------------------------------------------------*/
void DDS_Init (void) {

  RCC->AHB1ENR    |= ((1UL <<  4));          /* Enable GPIOE clock */

  GPIOE->MODER    &= ~((3UL << 2* 3) |
                       (3UL << 2* 4) |
                       (3UL << 2* 5) );   /* PE.0,3-4 are outputs */
  GPIOE->MODER    |=  ((1UL << 2* 3) |
                       (1UL << 2* 4) |
                       (1UL << 2* 5) );
  GPIOE->OTYPER   &= ~((1UL <<    3) |
                       (1UL <<    4) |
                       (1UL <<    5) );   /* PE.0,3-4 are output Push-Pull */
  GPIOE->OSPEEDR  &= ~((3UL << 2* 3) |
                       (3UL << 2* 4) |
                       (3UL << 2* 5) );   /* PE.0,3-4 are 50MHz Fast Speed */
  GPIOE->OSPEEDR  |=  ((2UL << 2* 3) |
                       (2UL << 2* 4) |
                       (2UL << 2* 5) );
  GPIOE->PUPDR    &= ~((3UL << 2* 3) |
                       (3UL << 2* 4) |
                       (3UL << 2* 5) );   /* PE.0,3-4 are Pull up */
  GPIOE->PUPDR    |=  ((1UL << 2* 3) |
                       (1UL << 2* 4) |
                       (1UL << 2* 5) );
}

void Pulse_Clock() {
  GPIOE->ODR |= (1 << CLOCK);
  Delay(1);
  GPIOE->ODR &= ~(1 << CLOCK);
}

void Pulse_Frequency() {
  GPIOE->ODR |= (1 << LOAD);
  Delay(1);
  GPIOE->ODR &= ~(1 << LOAD);
}

void Data_Low() {
  GPIOE->ODR &= ~(1 << DATA);
}

void DDS_Write_Data(int input_data) {
  GPIOE->ODR |= (input_data << DATA);
}

/*-------------------------------------------------------------------------
  Function that set the DDS output to a default value
 *-------------------------------------------------------------------------*/
void DDS_Default_Init (void) {

  int i = 0;
  int Default_Data[40] = {0,0,0,1,1,1,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}; //1KHz

  Pulse_Clock();
  Delay(1);
```

```
78      Pulse_Frequency();
79      Delay(1);
80
81      // Send the data array 1 bit at a time to the DDS
82      for(i = 0; i <40; i++){
83        Data_Low();
84        DDS_Write_Data(Default_Data[i]);
85        Delay(1);
86        Pulse_Clock();
87      }
88
89      Pulse_Frequency();
90      Delay(1);
91      Data_Low();
92
93    }
94    /*-------------------------------------------------------------------------
95      Function that sets the DDS frequency to a user provided value
96     *-------------------------------------------------------------------------*/
97    void DDS_Set (double frequency) {
98
99      int j = 0;
100     int k = 0;
101     int tuningWord = 0;
102     int Send_Data[40];
103
104     // Calculate the new tuning word
105     tuningWord = (int) ((frequency * pow(2, 32))/DDS_CLOCK);
106
107     // Construct the data array ready to be sent to DDS
108     for (j = 0; j < 40; j++) {
109       // calculate each array position by bitwise anding the tuning word with 1
110       Send_Data[j] = tuningWord & (1 << j) ? 1 : 0;
111     }
112
113     Pulse_Clock();
114     Delay(1);
115     Pulse_Frequency();
116     Delay(1);
117
118     // Send the data array 1 bit at a time to the DDS
119     for(k = 0; k <40; k++){
120       Data_Low();
121       DDS_Write_Data(Send_Data[k]);
122       Delay(1);
123       Pulse_Clock();
124     }
125
126     Pulse_Frequency();
127     Delay(1);
128     Data_Low();
129   }
130
```