```c
/*
 * peak_detect.c
 *
 *  Created on: Dec 12, 2018
 *      Author: Dominic Doty
 */

/* HEADER */
#include "peak_detect.h"

/* DEFINES AND STATIC DATA */
#define dBFS_LUT_COUNTS    {0,1,3,7,15,31,63,127,255,511,1023,2047,4095,8191,16383,32767}
#define dBFS_LUT_dB    {12700,9000,8100,7300,6700,6000,5400,4800,4200,3600,3000,2400,1800,1200,600,0}
#define dBFS_LUT_SLOPE {0,196804805,14745600,6553600,2457600,1433600,614400,307200,153600,76800,38400,19200,9600,4800,2400,1200}
#define dBFS_LUT_ENTRIES 16

static uint32_t dBFS_Counts[] = dBFS_LUT_COUNTS;
static uint32_t    dBFS_dB[] = dBFS_LUT_dB;
static uint32_t dBFS_Slope[] = dBFS_LUT_SLOPE;

/* FUNCTION DEFINITIONS */
uint16_t peak_output(volatile int16_t* buffer, uint8_t buffer_size, uint8_t decay_shift)
{
    // Calc Decay Number
    static uint16_t decay_number = 0;
    decay_number >>= decay_shift;

    // Find Max in Buffer
    uint16_t max = 0;
    for(volatile int16_t* ptr = &buffer[0]; ptr < &buffer[buffer_size]; ptr++)
    {
        if(abs(*ptr) > max)
        {
            max = abs(*ptr);
        }
    }

    if(max > decay_number)
    {
        decay_number = max;
    }

    return decay_number;
}

// Take a ADC Reading and Convert to 16 bit scale dBFS - note result is unsigned but all values should be presented as negative
int16_t dbfs_output(uint16_t input)
{
    uint32_t output = 0;

    // Find the LUT entry we need to look near
    uint8_t index = 0;
    uint16_t input_abs = abs(input);
    while(input)
    {
        input >>= 1;
        index++;
    }
    if(index == 0)
    {
        output = dBFS_dB[index];
    }
    else
    {
        // Interpolate
        uint32_t db_smaller = dBFS_dB[index - 1];
        uint32_t counts_smaller = dBFS_Counts[index - 1];
        uint32_t slope = dBFS_Slope[index];

        // Line
        output = db_smaller - ( ((input_abs - counts_smaller)*slope) / (2<<15) );
    }

    return (uint16_t)output;
}

void pretty_print(uint16_t sample, uint8_t scale_shift)
{
    uint16_t limit = ( sample >> scale_shift);
```

```
80      printf("%0*d>\n", limit, 0);
81  }
```