

# TM10P

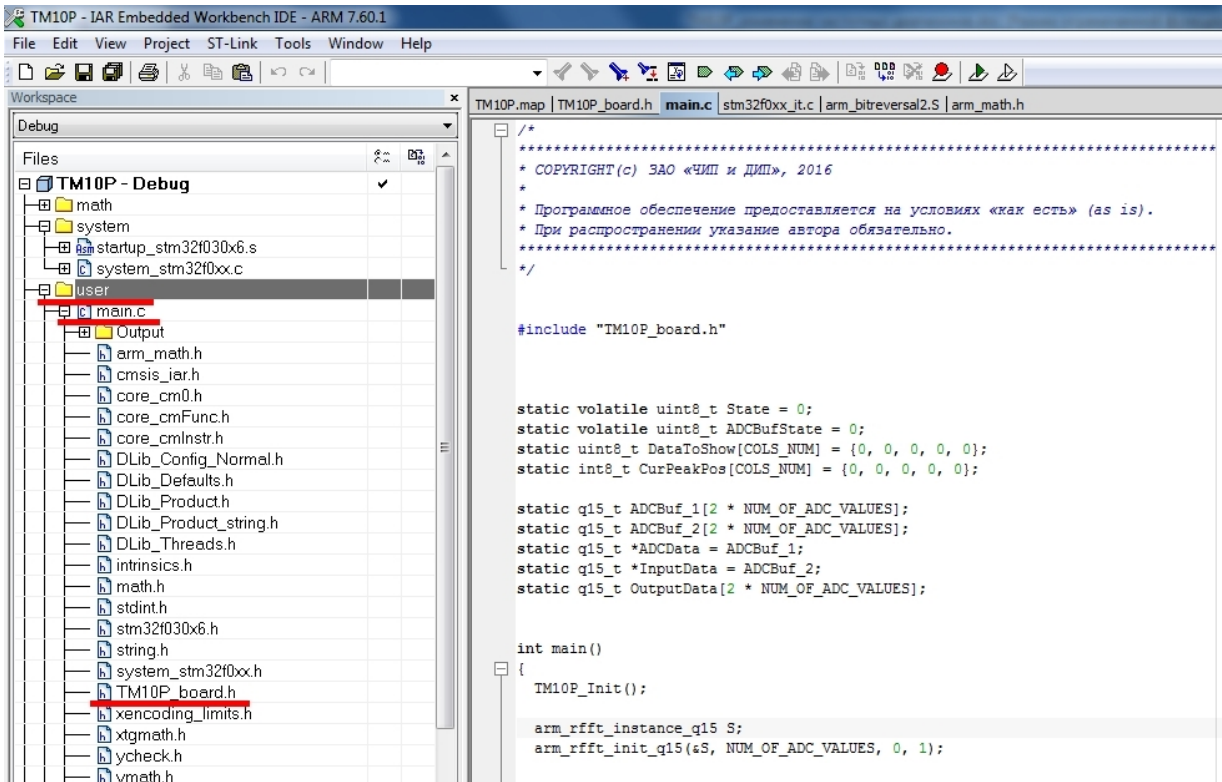
## Светодиодный 5-ти полосный анализатор спектра

### Изменение частотных диапазонов

По умолчанию полосы соответствуют следующим диапазонам частот:

- 1) от 625 Гц до 937.5 Гц;
- 2) от 1250 Гц до 1562.5 Гц;
- 3) от 1875 Гц до 3125 Гц;
- 4) от 3437.5 Гц до 6250 Гц;
- 5) от 6562.5 Гц до 19687.5 Гц.

Для изменения частот откройте проект TM10P в среде IAR Embedded Workbench. Разверните папку “User”. В папке “User” разверните файл “main.c”. Выполните двойной щелчок мыши на файле “TM10P\_board.h”.



The screenshot shows the IAR Embedded Workbench IDE interface. On the left, the 'Workspace' pane displays the project structure for 'TM10P - Debug'. The 'User' folder is expanded, and 'main.c' is selected. The main editor window shows the code for 'main.c', which includes 'TM10P\_board.h' and defines several static variables and the main function. The code is as follows:

```
/*
*****
* COPYRIGHT (c) ЗАО «ЧИП и ДИП», 2016
*
* Программное обеспечение предоставляется на условиях «как есть» (as is).
* При распространении указание автора обязательно.
*****
*/

#include "TM10P_board.h"

static volatile uint8_t State = 0;
static volatile uint8_t ADCBufState = 0;
static uint8_t DataToShow[COLS_NUM] = {0, 0, 0, 0, 0};
static int8_t CurPeakPos[COLS_NUM] = {0, 0, 0, 0, 0};

static q15_t ADCBuf_1[2 * NUM_OF_ADC_VALUES];
static q15_t ADCBuf_2[2 * NUM_OF_ADC_VALUES];
static q15_t *ADCData = ADCBuf_1;
static q15_t *InputData = ADCBuf_2;
static q15_t OutputData[2 * NUM_OF_ADC_VALUES];

int main()
{
    TM10P_Init();

    arm_rfft_instance_q15 S;
    arm_rfft_init_q15(&S, NUM_OF_ADC_VALUES, 0, 1);
```

Файл “TM10P\_board.h” откроется в окне справа. Определения для начала диапазонов частот обозначены как BAND\_1\_FIRST\_BIN ... BAND\_5\_FIRST\_BIN; определения для конца диапазонов частот обозначены как BAND\_1\_LAST\_BIN ... BAND\_5\_LAST\_BIN. Номера 1 ... 5 соответствуют номерам полос устройства.

Частота дискретизации устройства составляет 40 кГц, количество точек в преобразовании Фурье равно 128. Частотное разрешение  $40 \text{ (кГц)} / 128 = 312,5 \text{ (Гц)}$ . Частота, которую нужно отобразить на индикаторе определяется как  $312,5 \text{ (Гц)} * N$ ; где N – номер гармоники от 1 до  $128/2 = 64$ .

Таким образом, определения BAND\_1\_FIRST\_BIN ... BAND\_5\_FIRST\_BIN и BAND\_1\_LAST\_BIN ... BAND\_5\_LAST\_BIN задают номера гармоник. Например, если для BAND\_3\_FIRST\_BIN задано 7, а для BAND\_3\_LAST\_BIN – 17, то на третьей полосе устройства будет отображаться диапазон от  $312,5 * 7 = 2187,5 \text{ (Гц)}$  до  $312,5 * 17 = 5312,5 \text{ (Гц)}$ . Подставьте значения для BAND\_1\_FIRST\_BIN ... BAND\_5\_FIRST\_BIN и BAND\_1\_LAST\_BIN ... BAND\_5\_LAST\_BIN, соответствующие нужным диапазонам частот.



The screenshot shows the IAR Embedded Workbench IDE interface. On the left, the 'Workspace' pane displays the project structure for 'TM10P - Debug', including folders like 'math', 'system', and 'user', and files like 'main.c' and 'stm32f0xx\_it.c'. The main editor window shows the content of 'TM10P\_board.h', which contains several preprocessor definitions for frequency bands and ADC values. A red box highlights the definitions for 'BAND\_1\_LAST\_BIN' through 'BAND\_5\_LAST\_BIN' and their corresponding 'FIRST\_BIN' values.

```

#define SINGLE_LED_MODE (1 << 5)
#define ADC_BUFFER_FULL (1 << 0)

#define NUM_OF_ADC_VALUES 128

#define THRESHOLD_COEF 450

//соответствие частот столбцам индикатора
//частота = (частота дискретизации / количество выборок АЦП) * индекс
#define COL_1_FREQ 2 // (40000 / 128) * 2 = 625 (Гц)
#define COL_2_FREQ 6 // 1875 (Гц)
#define COL_3_FREQ 10 // 3125 (Гц)
#define COL_4_FREQ 16 // 5000 (Гц)
#define COL_5_FREQ 24 // 7500 (Гц)

#define BAND_1_LAST_BIN 3
#define BAND_2_LAST_BIN 5
#define BAND_3_LAST_BIN 10
#define BAND_4_LAST_BIN 20
#define BAND_5_LAST_BIN 63

#define BAND_1_FIRST_BIN 2
#define BAND_2_FIRST_BIN (BAND_1_LAST_BIN + 1)
#define BAND_3_FIRST_BIN (BAND_2_LAST_BIN + 1)
#define BAND_4_FIRST_BIN (BAND_3_LAST_BIN + 1)
#define BAND_5_FIRST_BIN (BAND_4_LAST_BIN + 1)

extern const float32_t WindowCoef[128];

void TM10P_Init(void);

void TIME_BASE_TIMER_ISR(void);

```

Отображение диапазонов частот на светодиодных полосах можно корректировать. Для этого выполните двойной щелчок мыши на файле “main.c”. Файл “ main.c ” откроется в окне справа.

The screenshot shows the 'main.c' file in the IDE. The code defines an array of coefficients for frequency bands. The line `float32_t BandCoef[COLS_NUM] = {1.0f, 0.25f, 0.25f, 0.025f, 0.07f};` is highlighted with a red line, indicating the values used for frequency correction.

```

//Для отображения диапазонов частот
float32_t BandCoef[COLS_NUM] = {1.0f, 0.25f, 0.25f, 0.025f, 0.07f};
uint8_t BandsStart[COLS_NUM] = {BAND_1_FIRST_BIN, BAND_2_FIRST_BIN, BAND_3_FIRST_BIN, BAND_4_FIRST_BIN, BAND_5_FIRST_BIN};
uint8_t BandsEnd[COLS_NUM] = {BAND_1_LAST_BIN, BAND_2_LAST_BIN, BAND_3_LAST_BIN, BAND_4_LAST_BIN, BAND_5_LAST_BIN};

uint32_t BandMagSum = 0;

for (uint8_t j = BandsStart[i]; j <= BandsEnd[i]; j++)
    BandMagSum += OutputData[j];

NewData[i] = (uint8_t)(BandMagSum / (THRESHOLD_COEF * BandCoef[i]));

```

В строке `float32_t BandCoef[COLS_NUM] = {1.0f, 0.25f, 0.25f, 0.025f, 0.07f};` задаются коэффициенты отображения частот для каждой полосы: 1.0f – для первой полосы, 0.07f – для пятой. Значение 1.0f означает, что отображение частоты происходит без изменения. Если значение больше 1.0f – отображение частоты корректируется в сторону уменьшения. Если значение меньше 1.0f – отображение частоты корректируется в сторону увеличения.

