РАБОТА С ПОРТОМ VPOUT МИКРОСХЕМЫ 1892ВМ10Я НА ПРИМЕРЕ МОДУЛЯ ДИСПЛЕЯ MC-LCD-TOUCH

1. ВВЕДЕНИЕ

Документ описывает порядок работы с дисплейным модулем MC-LCD-TOUCH, подключенным к порту VPOUT микросхемы 1892ВМ10Я на отладочном модуле NVCom-02TEM-3U.

Проект MCStudio 3M для данного примера доступен на официальном сайте multicore.ru в разделе «Техподдержка → Программное обеспечение → Примеры программирования».

2. ПОРТ ВЫВОДА ВИДЕОДАННЫХ VPOUT НА МИКРОСХЕМЕ 1892ВМ10Я

Порт вывода видеоданных VPOUT предназначен для вывода изображений по 16-разрядному параллельному интерфейсу. В данном документе рассматривается вывод изображения в формате 16-разрядного RGB (5R/6G/5B) на дисплейный модуль MC-LCD-TOUCH.

Более подробно порт VPOUT описан в руководстве пользователя на микросхему 1892ВМ10Я.

3. СВЕДЕНИЯ О MC-LCD-TOUCH

Модуль MC-LCD-TOUCH состоит из жидкокристаллического дисплея модели AT070TN92 и емкостной панели multi-touch, подключенной к контроллеру FT5x06.

Разрешение дисплея - 480x800 пикселей.

Дисплей AT070TN92 имеет 24-разрядную шину входных данных пикселей – R[7:0] G[7:0] B[7:0]. Для передачи видеоинформации в формате 16-разрядного RGB (5R/6G/5B), в модуле MC-LCD-TOUCH к выводам VD[15:0] подключены только старшие разряды выводов R[7:0], G[7:0], B[7:0]. Младшие разряды подключены к земле.

Таблица 3.1 Подключение AT070TN92 в составе модуля MC-LCD-TOUCH к 1892ВМ10Я

1892ВМ10Я	MC-LCD-TOUCH	AT070TN92
VDout[15:11]	VD[15:11]	R[7:3]
VDout [10:5]	VD[10:5]	G[7:2]
VDout [4:0]	VD[4:0]	B[7:3]

Дисплей AT070TN92 может работать в двух режимах - DE и SYNC.

Note 1: DE/SYNC mode select. Normally pull high.
When select DE mode, MODE="1", VS and HS must pull high.
When select SYNC mode, MODE= "0", DE must be grounded.

Рисунок 3.1 Информация о режимах работы из спецификации на дисплей AT070TN92 (Раздел 1, стр. 3)

Для выбора режима DE необходимо входы MODE, VSYNC(кадровая синхронизация), HSYNC(строчная синхронизация) подключить к «1». Вход DE будет использоваться для фиксации отображаемых пикселей.

Для выбора режима SYNC необходимо входы MODE и DE подключить к «0». Для фиксации начала/конца кадра и строки будут использоваться сигналы VSYNC и HSYNC. В таком режиме, помимо отображаемой области, дисплей имеет BLANK-области. Это временные интервалы после начала и перед концом фиксации кадра/строки, в которых данные на параллельной шине фиксируются, но не выводятся на дисплей. Эти области приведены на Рисунок 4.1.

В связи с тем, что в структуре порта VPOUT микросхемы 1892ВМ10Я нет сигнала Data Enable(DE), в модуле MC-LCD-TOUCH аппаратно установлен режим SYNC. На входах MODE и DE установлен низкий уровень напряжения.

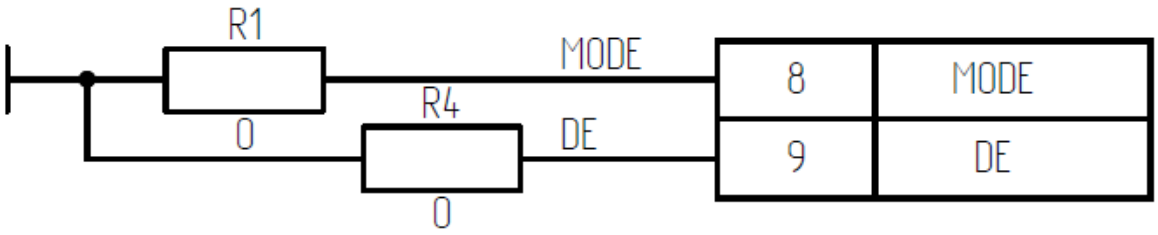


Рисунок 3.2 Схема подключения сигналов MODE и DE в модуле MC-LCD-TOUCH

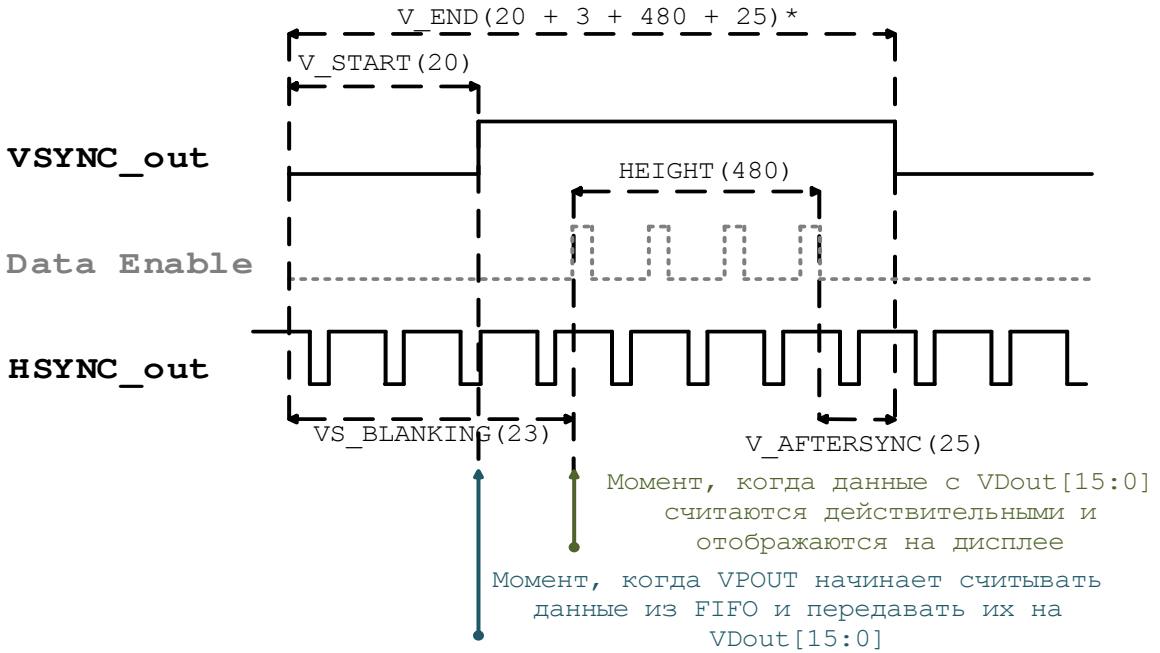
Для упрощения далее Data Enable используется как признак действительности данных. Однако физически постоянно DE = 0.

Для определения неактивной части данных вводятся параметры: VS_BLANKING/HS_BLANKING. Они обозначают количество периодов строчной/ синхронизации соответственно, перед тем, как данные станут действительными.

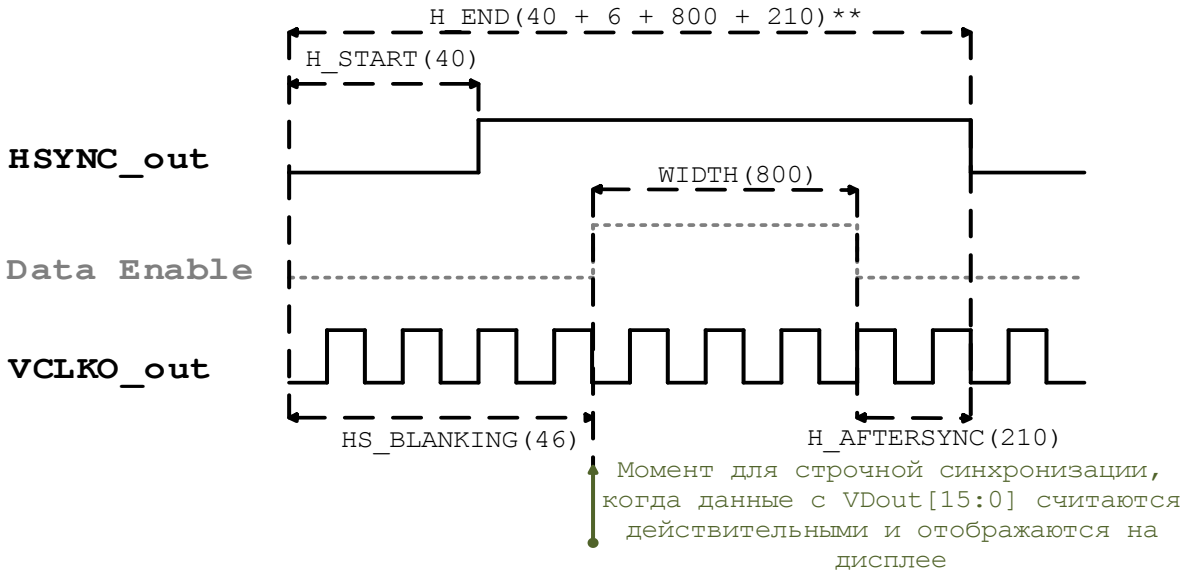
Для дисплея AT070TN92:

- VS_BLANKING = 23;
- HS_BLANKING = 46.

4. ОБЩАЯ ВРЕМЕННАЯ ДИАГРАММА И ПОЯСНЕНИЯ К НЕЙ



**Значения указаны в размерности периода сигнала HSYNC_out, который в свою очередь зависит от периода VCLKO_out*



***Значения указаны в размерности периода сигнала VCLKO_out, который в свою очередь зависит от значения регистра DIV*
TVCLKO_out = THCLK(DIV + 1), где THCLK – Период системной частоты работы CPU*

Рисунок 4.1 Временные диаграммы сигналов синхронизации

Таблица 4.1 Основные параметры сигналов синхронизации

Название	Назначение
VCLKO_out	Сигнал пиксельной синхронизации
HSYNC_out	Сигнал строчной синхронизации
VSYNC_out	Сигнал кадровой синхронизации
Data Enable	Признак действительности данных
H_START	Количество тактов VCLKO_out, при которых HSYNC_out = 0
H_END	Общий период сигнала HSYNC_out, в размерности VCLKO_out
HS_BLANKING	Длительность задержки Data Enable в строчной синхронизации
H_AFTERSYNC	Задержка HSYNC_out, после окончания активной части строки
V_START	Количество тактов HSYNC_out, при которых VSYNC_out = 0
V_END	Общий период сигнала VSYNC_out, в размерности HSYNC_out
VS_BLANKING	Длительность задержки Data Enable в кадровой синхронизации
V_AFTERSYNC	Задержка VSYNC_out, после окончания активной части кадра

Значения V_START, V_END, H_START, H_END определяются записью в соответствующие регистры VPOUT. В соответствии с этими значениями изменяются временные интервалы низкого и высокого уровня сигналов VSYNC и HSYNC. В документации на дисплей AT070TN92 указаны ограничения на эти интервалы.

Таблица 4.2 Ограничения параметров для сигналов синхронизации в AT070TN92

Интервал	Минимум	Рекомендуемое среднее значение	Максимум
H_START	1	-	40
H_END	862	1056	1200
HS_BLANKING	46	46	46
H_AFTERSYNC	16	210	354
V_START	1	-	20
V_END	510	525	650
VS_BLANKING	23	23	23
V_AFTERSYNC	7	22	147

Таким образом, значения V_START, V_END/H_START, H_END влияют на момент начала и конца передачи кадра/строки. Следовательно, меняется количество пикселей, которые будут считаны портом VPOUT и переданы на дисплей, но никак не отобразятся. Это необходимо учитывать при конфигурации DMA и записи кадра в память микросхемы.

5. ФОРМИРОВАНИЕ КАДРА В ПАМЯТИ

Кадр для передачи в VPOUT формируется из файла в формате BMP. Программа формирует в памяти видеобуфер размером 508x1016 пикселей. Это больше, чем разрешение дисплея 480x800, так как дополнительно в кадре учтены «пустые» области, не выводимые на экран.

Видимая часть изображения вычитывается из файла BMP, загруженного в память, после анализа BMP-заголовка.

```
#define PICT_ADDR 0x80200000

void Get_pict(unsigned int _frame_addr){
    //Проверка на соответствие файла формату BMP не осуществляется
    //Необходимые параметры загружаемого файла
    //Разрешение HEIGHT*WIDTH - 480x800
    //Глубина цвета 16 [бит/пиксель]
    //Без сжатия
    //Смещение изображения от начала файла находится
    //в заголовке файла - по адресу (начало_файла + 0xA)
    //Порядок следования пикселей в формате BMP:  СЛЕВА НАПРАВО И
СНИЗУ ВВЕРХ
    //Смещение изображения от начала файла
    unsigned int bmp_header_offset = *(unsigned short*)(PICT_ADDR +
0xA);
    //Указатель на начало картинки + смещение
    unsigned short *pict_data = (unsigned short*)(PICT_ADDR +
bmp_header_offset);
    //Смещение указателя чтения из загруженной картинки на начало
предпоследней строки
    pict_data += ((HEIGHT-1)*WIDTH);
    //Указатель на начало первого действительного пикселя
    unsigned short* ptr_x = (unsigned short*) (_frame_addr) +
FRAME_OFFSET;
    unsigned int x, y;
    //Кадр
    for (y = 0; y < HEIGHT; y++){//480
        for (x = 0; x < WIDTH ; x++)//800
            *ptr_x++ = *pict_data++;
        //Перемещение на начало предыдущей строки
        pict_data -= 2*WIDTH;
        //Отступ при записи следующей строки пикселей
        ptr_x += (HS_BLANKING - H_START + H_AFTERSYNC);}}
```

Перед записью кадра необходимо сделать отступ на величину FRAME_OFFSET. А после записи строки, необходимо сделать отступ равный по величине (HS_BLANKING - H_START + H_AFTERSYNC).

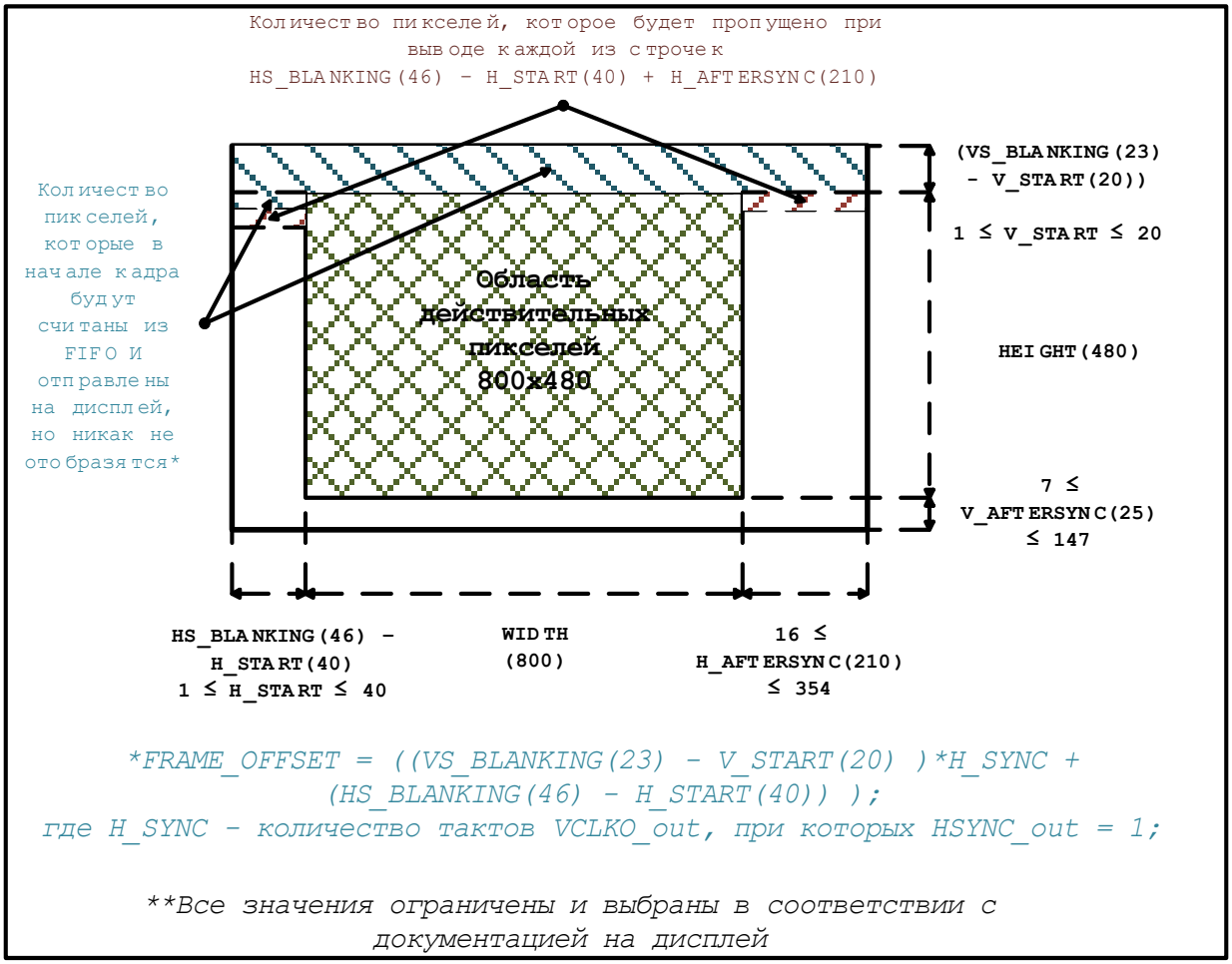


Рисунок 5.1 Схематичное изображение активной и неактивной части кадра

Файл BMP загружается в память микросхемы с помощью функции отладчика Load memory в среде разработки MCStudio 3M.

Для автоматического использования функции Load Memory необходимо:

- в проекте во время отладки установить точку загрузки «Load» в том месте программы, после которого будет необходим загруженный файл. (Например: Рисунок 5.2, точка загрузки «Load» установлена в месте вызова функции Get_pict(unsigned int _frame_addr));

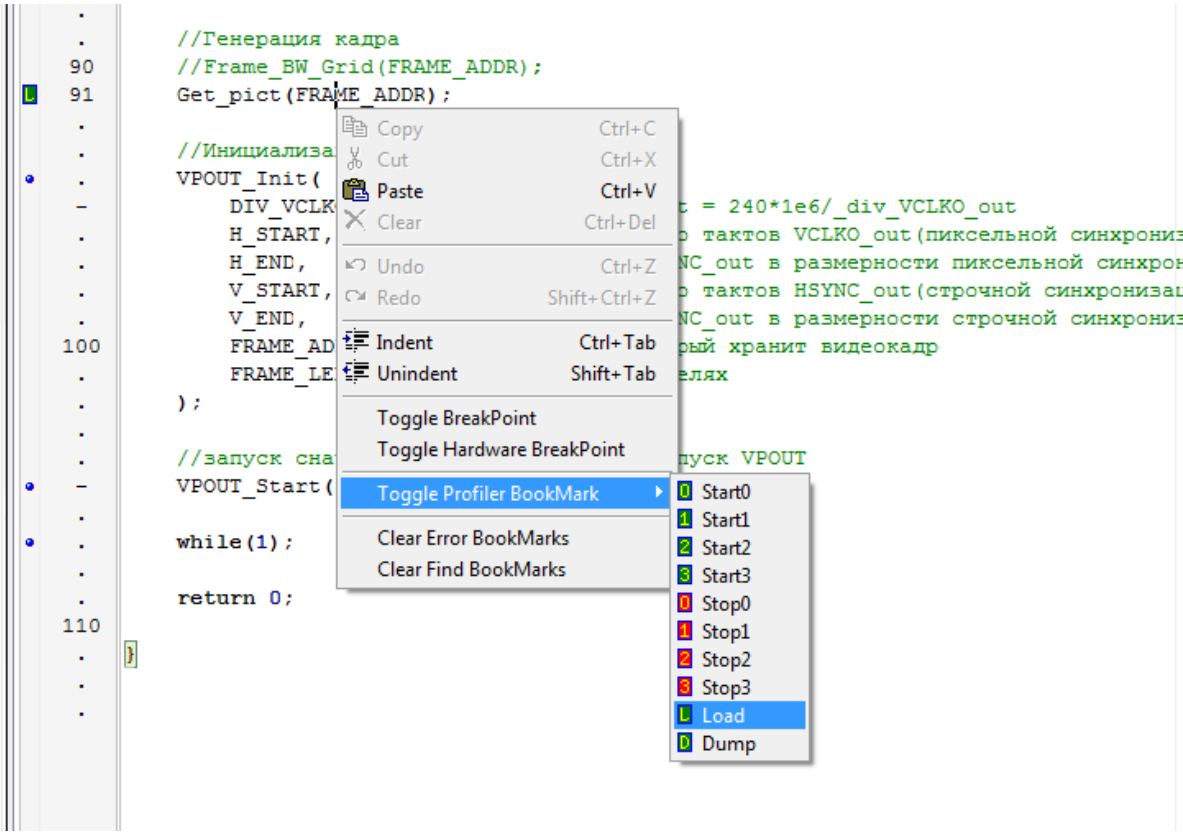


Рисунок 5.2 Установка точки загрузки для функции отладчика Load Memory

- в проекте во время отладки установить входные параметры Load Memory
  - путь к файлу;
  - адрес в микросхеме;
  - размер файла.

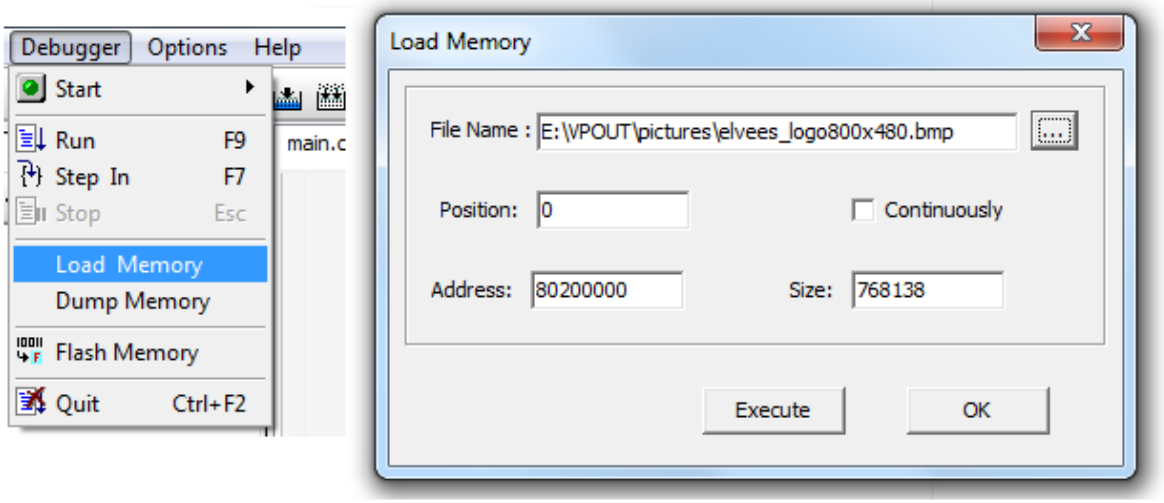


Рисунок 5.3 Конфигурация функции Load Memory в MCStudio 3M.

6. ИНИЦИАЛИЗАЦИЯ ПОРТА

Для начала работы порта VPOUT необходимо:

- выполнить его очистку записью 1 в поле CLR регистра CSR_VPOUT;

```
    CSR_VPOUT = (1<<31);
```

- настроить режим работы порта. Для работы с дисплеем AT070TN92 порт настраивается таким образом, чтобы он сам генерировал синхроимпульсы VCLKO, HSYNC, VSYNC;

```
    CSR_VPOUT = (1<<22) | (1<<21) | (1<<20);
```

- установить значения параметров H_START, H_END, V_START, V_END, согласно временной диаграмме на Рисунок 4.1 и ограничений из Таблица 4.2

```
    HstartHend_VPOUT = ((_hstart & 0x3FFF) << 16) | (_hend & 0xFFF);
    VstartVend_VPOUT = ((_vstart & 0x3FFF) << 16) | (_vend & 0xFFF);
```

- установить значение делителя частоты.

Для надежной передачи данных через порт VPOUT следует соблюдать ограничение, согласно которому период выходной частоты порта VCLKO должен превышать период частоты ядра не менее чем в 6 раз в том случае, если передаваемые данные фиксируются на приемной стороне по заднему фронту VCLKO (DIV≥5).

```
    DIV_VPOUT = _div_VCLKO_out;
```

В дисплее AT070TN92 данные регистрируются по заднему фронту, значит актуально ограничение DIV ≥ 5. Частота сигнала пиксельной синхронизации VCLKO_out = HCLK/(DIV + 1), где HCLK – системная частота работы CPU.

После этого порт все еще будет находиться в состоянии останова. Бит RUN, 30 бит регистра CSR_VPOUT, при начальной инициализации не устанавливается в «1». Так как, перед запуском порта VPOUT, необходимо запустить DMA-передачу.

```
//Запуск порта
    CSR_VPOUT |= (1<<30);
```

После запуска порт VPOUT будет генерировать сигналы кадровой(VSYNC), строчной(HSYNC) и пиксельной(VCLKO) синхронизации и будет передавать данные на дисплей в соответствии с временной диаграммой на Рисунок 4.1.

7. НАСТРОЙКА ЦЕПОЧКИ DMA ДЛЯ VPOUT

Данные в FIFO порта VPOUT могут быть записаны как со стороны CPU-ядра, так и через DMA. Запись через CPU займет почти все процессорное время, поэтому для передачи кадра необходимо использовать DMA. Один DMA-обмен позволяет передать до 65535 слов, то есть, 524288 байт. Чаще всего размер кадра - больше этого числа. При работе с дисплеем AT070TN92 размер видимой части кадра составляет 480x800 пикселей. Пиксель кодируется двумя байтами, то есть размер видимой части кадра составляет 768000 байт. С учетом неотображаемой части кадра его размер составляет: ((VS_BLANKING(26) – V_START(20)) + HEIGHT(480) + V_AFTERSYNC(210)) * ((HS_BLANKING(46) – H_START(40)) + WIDTH(800) + H_AFTERSYNC(210)) = V_SYNC(508)*H_SYNC(1016) = 516128 пикселей, то есть 1032256 байт. Для передачи кадра необходимо использовать цепочку DMA из двух обменов.

Особенность работы DMA VPOUT такова, что запуск следующей DMA-цепочки должен происходить пока VSYNC = «0»

При высоких уровнях HSYNC, VSYNC порт читает данные из FIFO независимо от того, есть они там или нет, и переключает указатель чтения. По концу кадра все указатели сбрасываются. При передаче изображения необходимо обеспечить постоянную подкачку данных из памяти микросхемы в FIFO порта VPOUT таким образом, чтобы FIFO никогда не успевало опустошаться.

С учетом этого, работа связки DMA-VPOUT должна быть организована в следующем порядке:

1. инициализация регистров порта VPOUT. 30 бит регистра CSR VPOUT отвечает за запуск порта, его необходимо установить в «0», чтобы порт находился в состоянии останова;
2. инициализация регистров DMA. Регистр RUN первого звена необходимо установить в «0», тогда DMA не начнет передачу, пока бит RUN не будет установлен в «1»;
3. запуск DMA;
4. запуск порта VPOUT;
5. ожидание прерывания по окончанию передачи DMA. Сброс битов DONE и END регистра CSR соответствующего канала DMA;
6. ожидание прерывания от VPOUT по окончанию кадра. В данный момент выполняется условие VSYNC = «0». Следовательно, необходимо снова запустить DMA, записью единицы в бит RUN регистра CSR. После повторного запуска вернуться к пункту 5.

Размер кадра с учетом передачи недействительных пикселей:

```
#define V_SYNC (VS_BLANKING - V_START + HEIGHT + V_AFTERSYNC)
#define H_SYNC (HS_BLANKING - H_START + WIDTH + H_AFTERSYNC)
//Видеоданные поступают на выход по положительному фронту сигнала
VCLKO_out
//при наличии одновременно активных (высоких) уровней HSYNC_out и
VSYNC_out
#define FRAME_LEN (H_SYNC)*(V_SYNC)          //Размер кадра в пикселях
```

Подробное описание параметров приведено в разделе «Общая временная диаграмма и пояснения к ней»: Рисунок 4.1, Таблица 4.1

_frame_len – количество пикселей, а каждый пиксель занимает два байта. В функцию конфигурации DMA передается смещение адреса относительно начала данных. Размер кадра в байтах frame_byte_size = _frame_len*0x2.

```
//Память для цепочки DMA
ChainDmaPort chain[2];
    //Размер кадра в байтах, так как каждый пиксель занимает 2 байта
    //(С учетом неотображаемых пикселей)
    unsigned int frame_byte_size = _frame_len*0x2;
//Инициализация цепочки
    DMA_Configure_chain(
        &chain[0],
        &chain[1],
        0, 0, 1, //RUN, IM, CHEN
        (unsigned int * )_frame_addr, //Начало кадра и до половины
        frame_byte_size/2);

    DMA_Configure_chain(
        &chain[1],
        &chain[0],
        1, 1, 1, //RUN, IM, CHEN
        (unsigned int * )( frame_addr + (frame_byte_size/2)),//Начало
кадра + половина
        frame_byte_size/2); //До конца кадра
//Загрузка первого звена в регистры DMA VPOUT
    DMA_Load_chain(&chain[0]);
```

8. ПРИНЦИП ПЕРЕДАЧИ КАДРА ИЗ ПАМЯТИ В ПОРТ. ПРЕРЫВАНИЯ.

При передаче данных по окончании каждого кадра (по фронту спада сигнала VSYNC) происходит принудительный сброс указателей FIFO. Таким образом, установленный в DMA порта заказ на передачу данных должен быть не более чем на 1 кадр, и каждый следующий кадр должен производиться новый запуск DMA. Для этого настраивается прерывание по концу кадра:

```
// Настройка прерываний для VPOUT и DMA_VPOUT
    SetCP0_Status((1<<10) | (1));// Разрешаем прерывания вообще - CP0.Status[0] = 1 и прерывания от VPOUT и DMA VPOUT - CP0.Status[10] = 1 IM[7:2]= 10 - прерывания, объединенные по ИЛИ в псевдорегистре QSTR0;
    MASKR0 |= (1<<19)|(1<<18);// Разрешаем прерывания от DMA VPOUT - MASKR0[19] = 1; MASKR0[18] = 1 – прерывания от DMA VPOUT
// Вектора прерывания размещаются во внутренней памяти CRAM (адреса типа 0xB800_0000);
    CSR |= (1<<1);
```

А затем разрешается прерывание самому порту VPOUT:

```
//Разрешаем прерывание VPOUT по концу кадра
    CSR_VPOUT |= (1<<26);
```

Далее добавить обработчик прерываний, который в приведенном случае состоит из двух частей – ассемблерный код (файл **irq.s**) и файл **irq_handler.c** основной функции обработчика.

irq.s – выполняет сохранение контекста и вызов функции **irq_handler()**, расположенной в **irq_handler.c**. Файл **irq.s** не содержит функций, специфичных для работы именно с портом VPOUT – аналогичные действия выполняются при обработке любого прерывания в RISC-ядре. Более подробно механизм обработки прерываний описан в документе «Применение процессоров серии «Мультикор». Обработка прерываний», доступном на сайте http://multicore.ru/.

Необходимо расположить файл **irq.s** по адресу, соответствующему вектору прерываний.

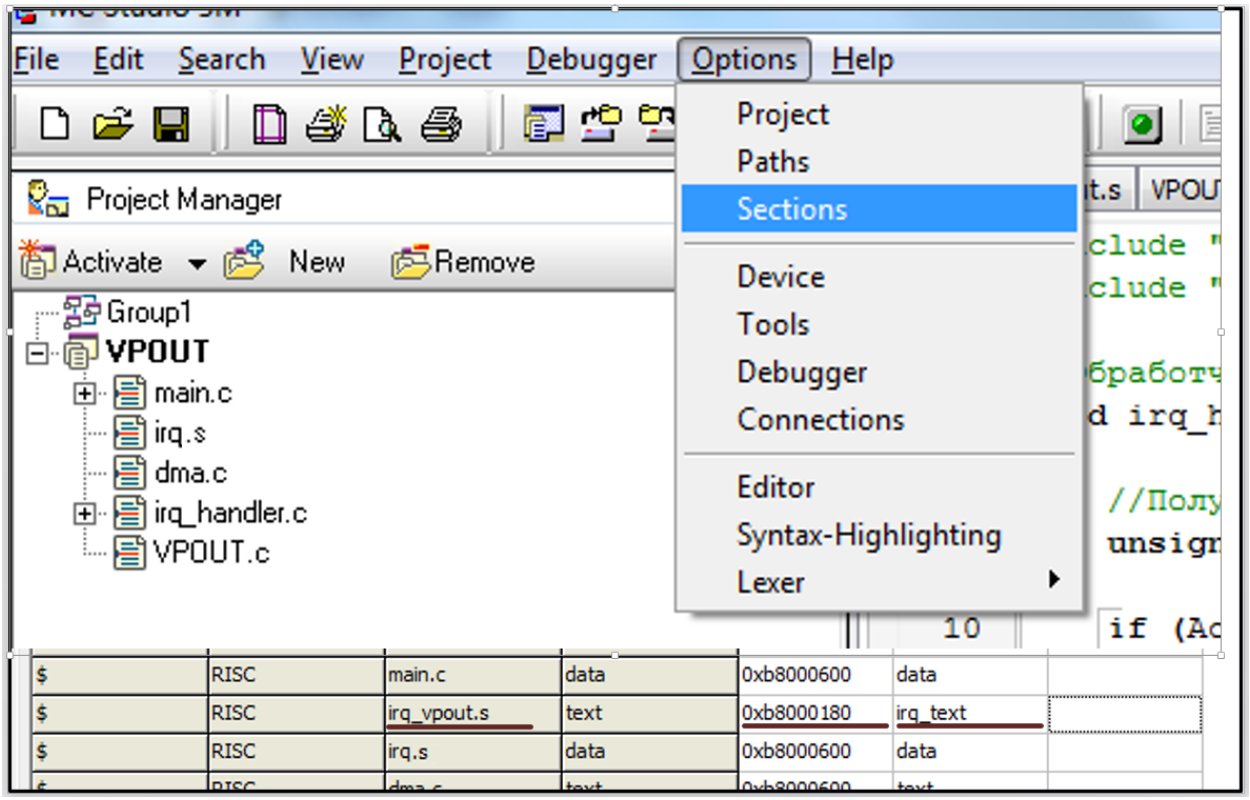


Рисунок 8.1 Смена секции для обработчика прерывания

При возникновении исключения процессор перейдет по адресу 0xb800_0180 и исполнит **irq.s**, который в свою очередь сохранит контекст и перейдет по метке в функцию **void irq_handler();**

```
// irq_handler.c
#include "memory_nvcom02_t.h"
//Обработчик прерывания
void irq_handler() {

    //Получаем код исключения
    unsigned int ActiveIRQ  = QSTR0 & MASKR0;

    if (ActiveIRQ & (1<<19)){//Если прерывания от канала DMA VPOUT по
передаче массива данных
        // Сбрасываем биты DONE и END
        CSR_VPOUT_CH &= ~((1<<15) | (1<<14));

    }
    if(ActiveIRQ & (1<<18)){
        if(((Frame_cnt_VPOUT & (1<<30)) == 0)  ){
            // Запускаем DMA для следующего кадра
            CSR_VPOUT_CH |= (1);
        }
    }}
```

Таким образом, в конце каждого кадра запускается новая передача DMA, а далее порт VPOUT считывает кадр из FIFO и передает его на дисплей.