I've been trying to make a motor controller out of STM32VL-Discovery (STM32F100RB). However I wanted just to check the waveforms and decided I could do without additional hardware. I'm emulating Hall signal with TIM2+DMA+GPIO. And emulation works great untill I try to initialize TIM3 or TIM4 as an interface timer. When I call TIM_Init() function, Hall emulation forms get corrupted. This happens immediately after entering the function, no actions done. And the optimization level setting of the compiler makes difference for how exactly the form is corrupted. It looks like something is messing with DMA preventing it from setting GPIO bits properly.
I run CooCox IDE 1.4.2 with arm-none-eabi-gcc-4_6.

Here's the code and illustrations:

```
01. int main(void)
02. {
03.     RCC_Configuration();
04.     //SysTick_Config(SystemCoreClock / 1);
05.     GPIO_Configuration();
06.     TIM_Configuration ();
07.     HallInterface_Configuration ();
08.
09.     while(1)
10.     {
11.
12.     }
13. }
```

next:

```
01. void RCC_Configuration(void)
02. {
03.     RCC_DeInit ();                    /* RCC system reset(for debug purpose)*/
04.     RCC_HSEConfig (RCC_HSE_ON);       /* Enable HSE                         */
05.
06.     /* Wait till HSE is ready                                               */
07.     while (RCC_GetFlagStatus(RCC_FLAG_HSERDY) == RESET);
08.
09.     RCC_HCLKConfig   (RCC_SYSCLK_Div1);   /* HCLK   = SYSCLK                */
10.     RCC_PCLK2Config  (RCC_HCLK_Div1);     /* PCLK2  = HCLK                  */
11.     RCC_PCLK1Config  (RCC_HCLK_Div2);     /* PCLK1  = HCLK/2                */
12.     RCC_ADCCLKConfig (RCC_PCLK2_Div4);    /* ADCCLK = PCLK2/4               */
13.
14.     /* PLLCLK = 8MHz * 3 = 72 MHz                                           */
15.     RCC_PLLConfig (0x00010000, RCC_PLLMul_3);
16.
17.     RCC_PLLCmd (ENABLE);                  /* Enable PLL                     */
18.
19.     /* Wait till PLL is ready                                               */
20.     while (RCC_GetFlagStatus(RCC_FLAG_PLLRDY) == RESET);
21.
22.     /* Select PLL as system clock source                                    */
23.     RCC_SYSCLKConfig (RCC_SYSCLKSource_PLLCLK);
24.
25.     /* Wait till PLL is used as system clock source                         */
26.     while (RCC_GetSYSCLKSource() != 0x08);
27.     RCC_SYSCLKConfig( RCC_SYSCLKSource_PLLCLK);
28. }
```

next:

```
01. void GPIO_Configuration(void)
02. {
03.     /* Initialize Leds mounted on STM32 board */
04.     GPIO_InitTypeDef  GPIO_InitStructure;
05.     /* Initialize LED which connected to PC6,9, Enable the Clock*/
06.     RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC, ENABLE);
07.
08.     GPIO_DeInit(GPIOC);
09.     /* Configure the GPIO_LED pin */
10.     GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9;
```

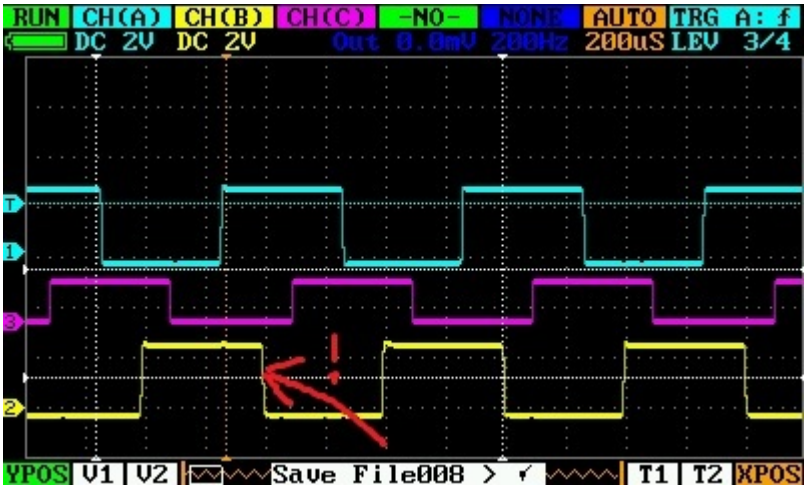
```
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOC, &GPIO_InitStructure);

    GPIO_ResetBits(GPIOC,GPIO_Pin_9);
}
```

next:

```
void TIM_Configuration (void)
{
    /**
     * TIM2 mocks Hall sensors output by consequently generating DMA request
     * to fetch from memory next state of three output pins
     * HallU - PC10
     * HallV - PC11
     * HallW - PC12
     */
    GPIO_InitTypeDef GPIO_InitStructure;
    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    DMA_InitTypeDef DMA_InitStructure;

    const uint32_t GPIO_PinsState[6] = {
            134222848, 402654208, 268438528,
            335546368, 67115008, 201330688  };

    uint16_t FullCycleFreq = 1240; //Hz, 6 steps, full electrical rotation
    uint16_t TimerFreq = FullCycleFreq * 6;
    uint16_t TimerPeriod = (uint16_t)(SystemCoreClock/(TimerFreq + 1));

    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC, ENABLE);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10 | GPIO_Pin_11 | GPIO_Pin_12;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOC, &GPIO_InitStructure);

    GPIOC->BSRR = (1<<25)+(1<<24)+(1<<11);

    RCC_AHBPeriphClockCmd(RCC_AHBPeriph_DMA1, ENABLE);

    DMA_DeInit(DMA1_Channel2);
    DMA_InitStructure.DMA_PeripheralBaseAddr = (u32) GPIOC_BSRR_Address;
    DMA_InitStructure.DMA_MemoryBaseAddr = (u32) GPIO_PinsState;
    DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralDST;

    DMA_InitStructure.DMA_BufferSize = 6;
    DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Disable;
    DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Enable;
    DMA_InitStructure.DMA_PeripheralDataSize = DMA_PeripheralDataSize_Word;
    DMA_InitStructure.DMA_MemoryDataSize = DMA_MemoryDataSize_Word;
    DMA_InitStructure.DMA_Mode = DMA_Mode_Circular;
    DMA_InitStructure.DMA_Priority = DMA_Priority_High;
    DMA_InitStructure.DMA_M2M = DMA_M2M_Disable;

    DMA_Init(DMA1_Channel2, &DMA_InitStructure); //TIM2_UP = Channel2

    DMA_Cmd(DMA1_Channel2, ENABLE);

    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);

    TIM_TimeBaseStructure.TIM_Period = TimerPeriod;
    TIM_TimeBaseStructure.TIM_Prescaler = 0;
    TIM_TimeBaseStructure.TIM_ClockDivision = 0;
    TIM_TimeBaseStructure.TIM_RepetitionCounter = 0;
    TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;
    TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);

```

```
60.
61.     TIM_DMACmd(TIM2, TIM_DMA_Update, ENABLE);
62.
63.     TIM_Cmd(TIM2, ENABLE);
64. }
```

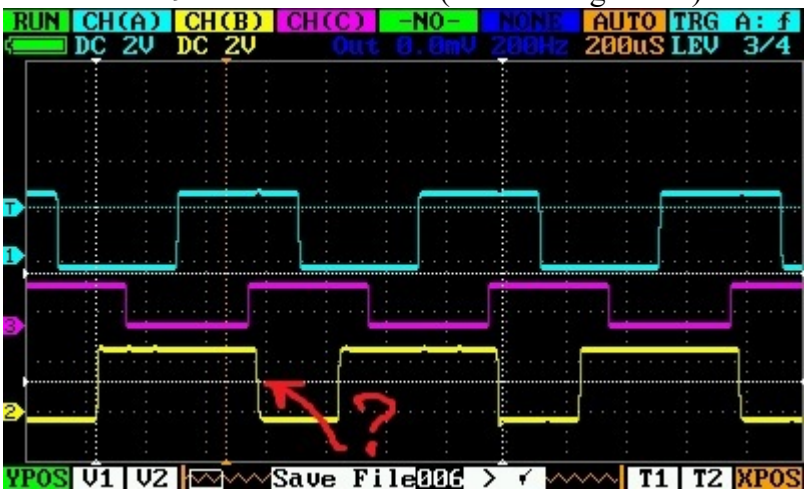
All the way until here everything is great. User LEDs are not powered and the waveform looks exactly as I need it:



Then something strange happen:

```
01. void InterfaceTimerSetup (void)
02. {
03.     TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure1;
04.     TIM_ICInitTypeDef TIM_ICInitStructure;
05.     TIM_OCInitTypeDef TIM_OCInitStructure;
06.     NVIC_InitTypeDef NVIC_InitStructure;
07.
08.     RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM4, ENABLE);
09.
10.     TIM_TimeBaseStructure1.TIM_Prescaler = 12;//126;
11.     TIM_TimeBaseStructure1.TIM_CounterMode = TIM_CounterMode_Down;//TIM_CounterMode_Up;
12.     TIM_TimeBaseStructure1.TIM_Period = 1;//65535;
13.     TIM_TimeBaseStructure1.TIM_ClockDivision = 1;//0;
14.     TIM_TimeBaseStructure1.TIM_RepetitionCounter = 1;//0;
15.     TIM_TimeBaseInit(TIM4, &TIM_TimeBaseStructure1);
16.
17. }
```

After this PC9 LED comes to life (constant high level) and the waveforme changes as follows:



Yellow line is PC11. I've tried to change interface timer to TIM3/4 - no matter. Changing optimization level changes more than one waveforms. Initing OC/IC changes more than one waveform. This is a complete

mistery to me - please please please help me out.
Thanks in advance