

Lösung Vorlesungsbegleitende Übungsaufgaben

-Mikrocomputertechnik-

Aufgabe 3: Interrupt Counter

Schreiben Sie ein Programm, dass bei Betätigung des USER Tasters einen Interrupt auslöst und den Wert einer Variable inkrementell erhöht. Der Wert der Variable soll in binärer Codierung durch die LEDs PE8 bis PE15 dargestellt werden. Kann der Wert nicht mehr dargestellt werden soll er zurückgesetzt werden.

Lösung:

The screenshot displays the STM32CubeMX software interface. On the left, the 'System Core' section shows the 'GPIO' peripheral selected. The 'Configuration' tab is active, showing the 'GPIO Mode and Configuration' for the 'GPIO' peripheral. The 'Search Signals' field is empty. The 'Show only Modified Pins' checkbox is unchecked. The table below shows the configuration for the GPIO pins:

Pin	Signal	GPIO n	GPIO o	GPIO	Maxim.	Fast M.	User L.	Modified
PA0	n/a	n/a	Extern...	No pull...	Low	n/a		<input type="checkbox"/>
PE8	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>
PE9	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>
PE10	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>
PE11	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>
PE12	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>
PE13	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>
PE14	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>
PE15	n/a	Low	Output...	No pull...	Low	n/a		<input type="checkbox"/>

Below the table, the 'PA0 Configuration' section shows the 'GPIO mode' set to 'External Interrupt Mode with Rising edge trigger detection'. The 'GPIO Pull-up/Pull-down' is set to 'No pull-up and no pull-down'. The 'User Label' field is empty.

The 'Pinout view' on the right shows the physical pins of the STM32F303VCTx LQFP100 package. The pins are labeled with their names (e.g., PE8, PE9, PE10, PE11, PE12, PE13, PE14, PE15) and their connections to the microcontroller. The pins are color-coded: yellow for VDD, VSS, and VBAT; blue for PA0, PA1, PA2, PA3, PA4, PA5, PA6, PA7, PA8, PA9, PA10, PA11, PA12, PA13, PA14, PA15; and green for PE8, PE9, PE10, PE11, PE12, PE13, PE14, PE15. The pins are arranged in a grid around the central microcontroller chip.

The screenshot shows the STM32CubeMX software interface. The 'NVIC Mode and Configuration' window is open, displaying the 'NVIC Interrupt Table'. The table lists various interrupts and their configurations. The 'EXTI line0 interrupt' is highlighted in blue. The 'Enabled' column for this interrupt is checked, and the 'Presemption Priority' is set to 0.

Interrupt	Enabled	Presemption Priority
Non maskable interrupt	<input checked="" type="checkbox"/>	0
Hard fault interrupt	<input checked="" type="checkbox"/>	0
Memory management fault	<input checked="" type="checkbox"/>	0
Pre-fetch fault, memory access fault	<input checked="" type="checkbox"/>	0
Undefined instruction or illegal state	<input checked="" type="checkbox"/>	0
System service call via SWI instruction	<input checked="" type="checkbox"/>	0
Debug monitor	<input checked="" type="checkbox"/>	0
Pendable request for system service	<input checked="" type="checkbox"/>	0
Time base: System tick timer	<input checked="" type="checkbox"/>	0
PVD interrupt through EXTI line16	<input type="checkbox"/>	0
Flash global interrupt	<input type="checkbox"/>	0
RCC global interrupt	<input type="checkbox"/>	0
EXTI line0 interrupt	<input checked="" type="checkbox"/>	0
Floating point unit interrupt	<input type="checkbox"/>	0

```
volatile static int InterruptCounter=0;

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if(GPIO_Pin == GPIO_PIN_0) //Auswertung von PA0
    {
        if(InterruptCounter>=255)
        {
            InterruptCounter=0;
        }
        else
        {
            InterruptCounter++;
        }
    }
}

int main(void)
{
    HAL_Init();

    SystemClock_Config();

    MX_GPIO_Init();
    while (1)
    {
        //Reset aller LEDs
        HAL_GPIO_WritePin(GPIOE, GPIO_PIN_8|GPIO_PIN_9|GPIO_PIN_10|GPIO_PIN_11|GPIO_PIN_12|
            GPIO_PIN_13|GPIO_PIN_14|GPIO_PIN_15, GPIO_PIN_RESET);

        //Setzen der jeweiligen LED
        if(InterruptCounter&1<<0)
        {
            HAL_GPIO_WritePin(GPIOE, GPIO_PIN_8, GPIO_PIN_SET);
        }
        if(InterruptCounter&1<<1)
        {
            HAL_GPIO_WritePin(GPIOE, GPIO_PIN_9, GPIO_PIN_SET);
        }
        if(InterruptCounter&1<<2)
        {
            HAL_GPIO_WritePin(GPIOE, GPIO_PIN_10, GPIO_PIN_SET);
        }
        if(InterruptCounter&1<<3)
        {

```

```
        HAL_GPIO_WritePin(GPIOE, GPIO_PIN_11, GPIO_PIN_SET);
    }
    if(InterruptCounter&1<<4)
    {
        HAL_GPIO_WritePin(GPIOE, GPIO_PIN_12, GPIO_PIN_SET);
    }
    if(InterruptCounter&1<<5)
    {
        HAL_GPIO_WritePin(GPIOE, GPIO_PIN_13, GPIO_PIN_SET);
    }
    if(InterruptCounter&1<<6)
    {
        HAL_GPIO_WritePin(GPIOE, GPIO_PIN_14, GPIO_PIN_SET);
    }
    if(InterruptCounter&1<<7)
    {
        HAL_GPIO_WritePin(GPIOE, GPIO_PIN_15, GPIO_PIN_SET);
    }

    //Kurzes Delay
    HAL_Delay(100);
}

static void MX_GPIO_Init(void)
{
    GPIO_InitTypeDef GPIO_InitStruct = {0};

    /* GPIO Ports Clock Enable */
    __HAL_RCC_GPIOA_CLK_ENABLE();
    __HAL_RCC_GPIOE_CLK_ENABLE();

    /*Configure GPIO pin Output Level */
    HAL_GPIO_WritePin(GPIOE, GPIO_PIN_8|GPIO_PIN_9|GPIO_PIN_10|GPIO_PIN_11
        |GPIO_PIN_12|GPIO_PIN_13|GPIO_PIN_14|GPIO_PIN_15, GPIO_PIN_RESET);

    /*Configure GPIO pin : PA0 */
    GPIO_InitStruct.Pin = GPIO_PIN_0;
    GPIO_InitStruct.Mode = GPIO_MODE_IT_RISING;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

    /*Configure GPIO pins : PE8 PE9 PE10 PE11
        PE12 PE13 PE14 PE15 */
    GPIO_InitStruct.Pin = GPIO_PIN_8|GPIO_PIN_9|GPIO_PIN_10|GPIO_PIN_11
        |GPIO_PIN_12|GPIO_PIN_13|GPIO_PIN_14|GPIO_PIN_15;
    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
    HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);

    /* EXTI interrupt init*/
    HAL_NVIC_SetPriority(EXTI0_IRQn, 0, 0);
    HAL_NVIC_EnableIRQ(EXTI0_IRQn);
}
```