

**LẬP TRÌNH STM32**

Bài 6: STM32 Timer chế độ Input Capture và Output Compare

POSTED ON 10/07/2020 BY KHUÊ NGUYỄN

Lập trình STM32 và CubeMX

Khuê Nguyễn Creator

STM32 Cube

Bài 6: Timer chế độ Input Capture và Output Compare

STM32 Timer có nhiều chế độ trong đó Input Capture (IC) và Output Compare (OC) là 2 chế độ được sử dụng rất nhiều trong lập trình nhúng. Trong bài này chúng ta sẽ tìm hiểu IC và OC là gì, cách cấu hình trên Cube MX và lập trình trên Keil C

Bài 6 trong Serie Học lập trình STM32 từ A tới Z

Mục Lục

- 1. STM32 Timer chế độ Input Capture
- 2. STM32 Timer chế độ output compare
- 3. Cấu hình STM32 Timer trong CubeMX
- 4. Lập trình STM32 Timer chế độ IC và OC
- 5. Kết
 - 5.1. Related posts:



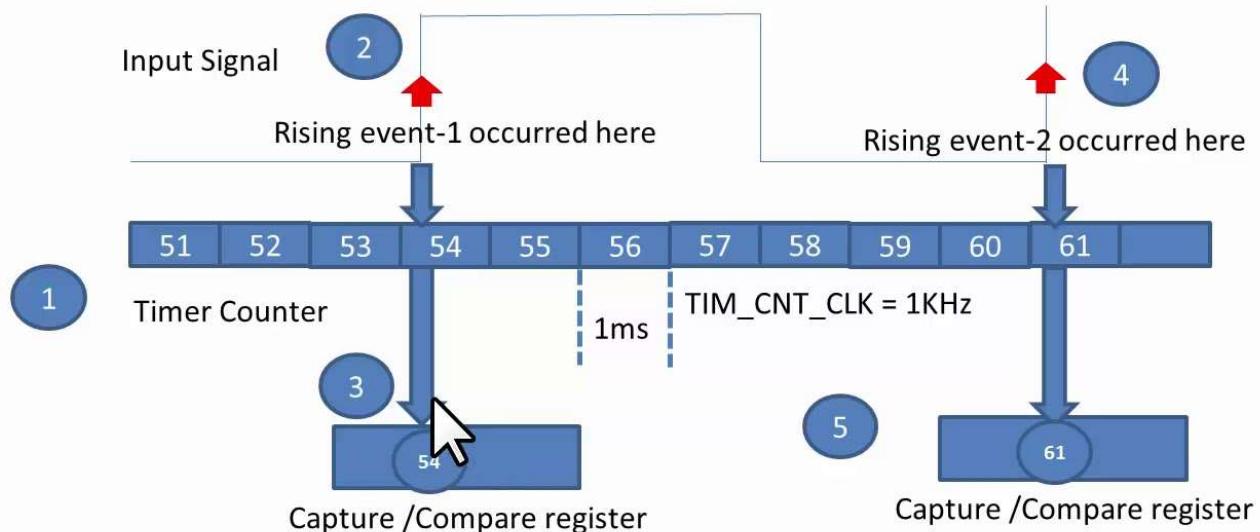
STM32 Timer chế độ Input Capture

Input Capture là chế độ bắt xườn đầu vào, thường ứng dụng trong các hoạt động đo tần số hoặc độ rộng xung.

Input Capture có 2 chế độ là bắt xung xườn lên(Rising) hoặc xướn xuống (Falling)

Nguyên lý hoạt động như sau:

- + Sau khi khởi động Timer , mỗi khi có xung xướn lên (hoặc xuống) tại đầu vào Channelx của Timer. Thanh ghi TIMx_CCRx (x là Timer số 1,2,3,4...) sẽ được nạp giá trị của thanh ghi Counter TIMx_CNT
- + Bit CC1IF được đặt lên 1(Cờ ngắt), Bit CC1OF sẽ được set lên 1 nếu 2 lần cờ liên tiếp cờ CC1IF được đặt lên 1 mà ko xóa. Nghĩa là mỗi khi có sự kiện xung lên sẽ có ngắt xảy ra, lập trình viên phải đọc giá trị lưu vào CCR1 sau đó xóa cờ CC1IF, nếu ko cờ CC1OF sẽ được bật lên báo tràn và dữ liệu sẽ ko được ghi vào nữa.
- + Một sự kiện ngắt được sinh ra nếu Bit CC1IE được đặt lên 1
- + Một sự kiện DMA sinh ra nếu Bit CC1DE được đặt lên 1



Bài 6: STM32 Timer chế độ Input Capture và Output Compare 52

Các bước khởi tạo IC như sau:

B1: Chọn nguồn xung đếm (internal, external, timer)

B2: Ghi dữ liệu vào Thanh ghi ARR và thanh ghi PSC

B4: Chọn bật Ngắt hoặc DMA bằng bit CCxIE hoặc CCxDE trong thanh ghi CR1

B5: Khởi chạy bộ đếm bằng Bit CEN của thanh ghi CR1

B6: Trong hàm ngắt đọc dữ liệu từ thanh ghi CCRx, xóa thanh ghi CNT về 0 khi có ngắt thứ 2 sinh ra, giá trị của thanh ghi CCRx là khoảng thời gian giữa 2 lần ngắt.

Chi tiết các bạn tham khảo mục 15.3.4 trong reference Manual

STM32 Timer chế độ output compare

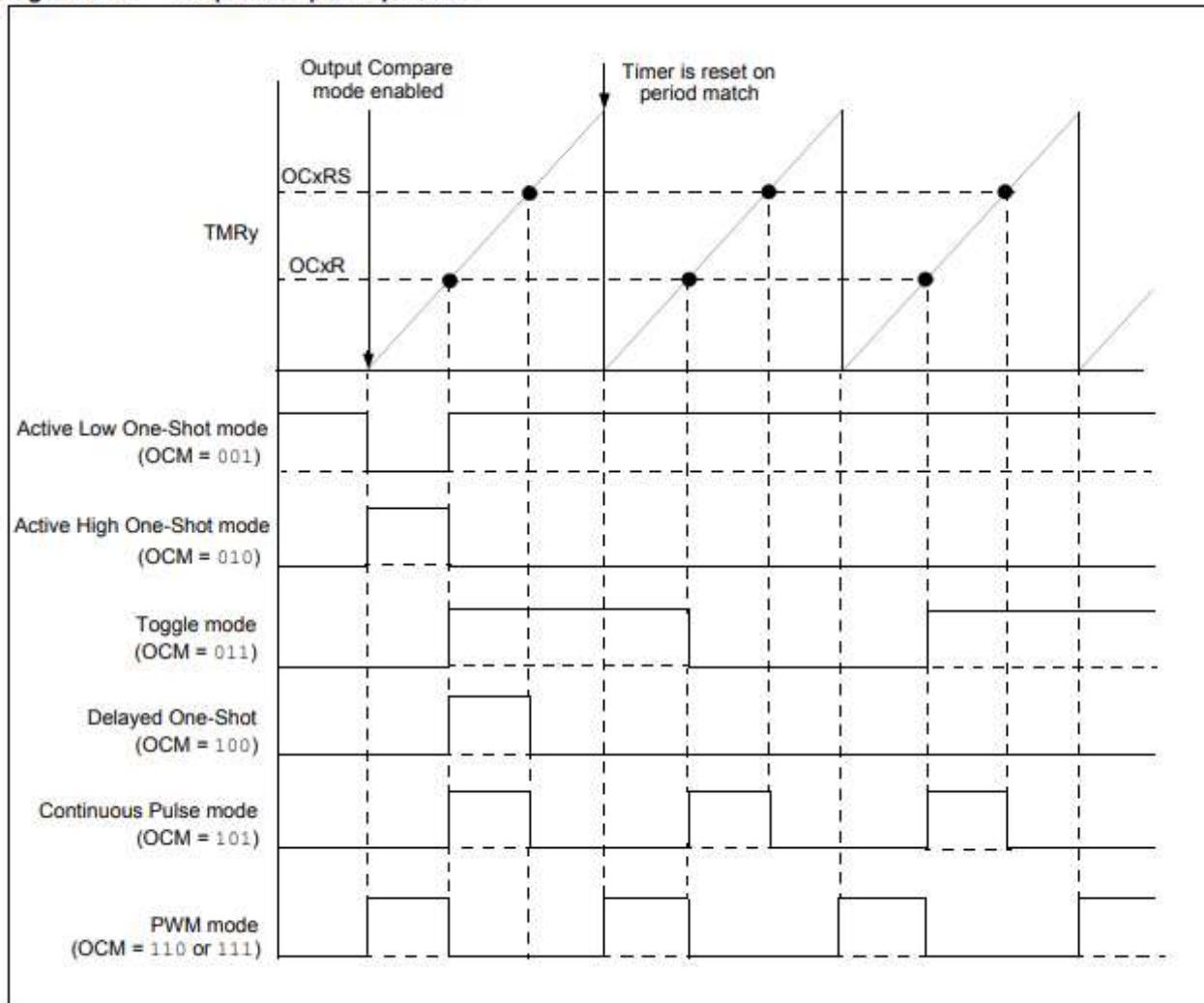
Chế độ output Compare là chế độ phát xung ra khi thời điểm được cài đặt, bằng thời điểm mà bộ đếm đang đếm lên/xuống.

Nguyên lý hoạt động: Khi bộ đếm, đếm đến giá trị được cài đặt trong thanh ghi CCR Timer sẽ:

- + Tạo ra một hoạt động tùy vào chế độ được chọn của bộ OC
- + Set cờ ngắt CCxIF lên 1
- + Tạo ra một ngắt hoặc DMA nếu bit CCxIE hoặc CCxDE được bật

Các chế độ làm việc của OC

Figure 13-2: Output Compare Operation



Bài 6: STM32 Timer chế độ Input Capture và Output Compare 53

Các bước khởi tạo OC như sau:

B1: Chọn xung Clock cho Timer

B2: Ghi dữ liệu vào thanh ghi ARR và thanh ghi CCR

B3: Set bit CCxIE hoặc CCxDE nếu bật ngắt hoặc DMA

B4: Chọn chế độ OC bằng 4 bit OCxM trong thanh ghi CCMR1

B5: Chạy bộ đếm bằng bit CEN của thanh ghi CR1

Chi tiết các bạn tham khảo mục 15.3.8 trong reference Manual

Cấu hình STM32 Timer trong CubeMX

Trong bài này, chúng ta sẽ cấu hình Channel 1 của Timer 2 là Input Capture,

Channel 1 của Timer 3 là Output Compare.

Mở phần mềm lên, chọn Khởi tạo trên MCU STM32F103C8, nhấn Start project

Trong Sys Debug chọn Serial Wire, nếu các bạn chưa biết các bước cơ bản này vui lòng đọc lại **Bài 3** nhé

Trong phần Timer chọn Timer 2 :

Clock Source – Internal Clock

Channel 1: Input Capture direct mode

Trong Parameters Settings chọn Prescaler là 8000 => mỗi lần đếm là 1ms (chi tiết trong bài 5)

Counter Period: 999 Chu kỳ tràn là 1000 ms. Vì mình sẽ đọc các xung có F > 1Hz

Auto Reload: Enable

Polarity Selection: Rising (Chọn bắt xung lên)

TIM2 Mode and Configuration

Mode

Slave Mode	Disable
Trigger Source	Disable
Clock Source	Internal Clock
Channel1	Input Capture direct mode
Channel2	Disable
Channel3	Disable
Channel4	Disable

Configuration

Reset Configuration

✓ NVIC Settings ✓ DMA Settings ✓ GPIO Settings
✓ Parameter Settings ✓ User Constants

🔍 Search (Ctrl+F) 🕒 🕒 ⓘ

- ✓ Counter Settings

Prescaler (PSC - 16 bits value)	8000
Counter Mode	Up
Counter Period (AutoReload Register)	999
Internal Clock Division (CKD)	No Division
auto-reload preload	Enable
- ✓ Trigger Output (TRGO) Parameters

Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection	Reset (UG bit from TIMx_EGR)
- ✓ Input Capture Channel 1

Polarity Selection	Rising Edge
IC Selection	Direct
Prescaler Division Ratio	No division
Input Filter (4 bits value)	0

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 54

Timer 3 Các bạn cấu hình như sau:

Internal Clock

Channel 1: Ouput Compare CH1

Prescaler 8000 => mỗi lần đếm là 1ms

Counter Period: 49 chu kì tràn là 50ms

Auto Reload: Enable

Mode: Toggle on Match: 24 (đổi trạng thái CH1 khi Counter đếm lên 24 => 25ms)

TIM3 Mode and Configuration

Mode

- Slave Mode: Disable
- Trigger Source: Disable
- Internal Clock
- Channel1: Output Compare CH1
- Channel2: Disable
- Channel3: Disable
- Channel4: Disable

Configuration

Parameter Settings

Configure the below parameters :

Parameter	Value
Prescaler (PSC - 16 bits value)	8000
Counter Mode	Up
Counter Period (AutoReload Register)	49
Internal Clock Division (CKD)	No Division
auto-reload preload	Enable
Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection	Reset (UG bit from TIMx_EGR)
Mode	Toggle on match
Pulse (16 bits value)	24
Output compare preload	Enable

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 55

NVIC Setting tick vào bật Ngắt cho Timer 2

NVIC Mode and Configuration

Configuration

NVIC | Code generation

Priority Group 4 bits for pre... ▾ Sort by Preemption Priority and Sub Priority

Search Sear... Show only enabled interrupts Force DMA channels Inte

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
Non maskable interrupt	<input checked="" type="checkbox"/>	0	0
Hard fault interrupt	<input checked="" type="checkbox"/>	0	0
Memory management fault	<input checked="" type="checkbox"/>	0	0
Prefetch fault, memory access fault	<input checked="" type="checkbox"/>	0	0
Undefined instruction or illegal state	<input checked="" type="checkbox"/>	0	0
System service call via SWI instruction	<input checked="" type="checkbox"/>	0	0
Debug monitor	<input checked="" type="checkbox"/>	0	0
Pendable request for system service	<input checked="" type="checkbox"/>	0	0
Tim			
PVD interrupt through EXTI line 16	<input type="checkbox"/>	0	0
Flash global interrupt	<input type="checkbox"/>	0	0
RCC global interrupt	<input type="checkbox"/>	0	0
TIM2 global interrupt	<input checked="" type="checkbox"/>	0	0
TIM3 global interrupt	<input type="checkbox"/>	0	0

Enabled Preemption Priority Sub Priority

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 56

Đặt tên, Chọn Toolchain và Gen code

Project Settings

Project Name
Bai6_IC_OC_Tim2

Project Location
D:\STM32 Cube Youtube

Application Structure
Basic Do not generate the main()

Toolchain Folder Location
D:\STM32 Cube Youtube\Bai6_IC_OC_Tim2\

Toolchain / IDE
MDK-ARM Min Version
V5 Generate Under Root

Linker Settings

Minimum Heap Size

Minimum Stack Size

Mcu and Firmware Package

Mcu Reference
STM32F103C8Tx

Firmware Package Name and Version
STM32Cube_FW_F1_V1.8.0

Use Default Firmware Location

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 57

Lập trình STM32 Timer chế độ IC và OC

Open Project Nhấn F7 để Build Code, Trong stm32f1xx_it.c các bạn tìm đến hàm HAL_TIM_IRQHandler(&htim2); và Goto define nó

The screenshot shows a code editor with the following code snippet:

```

201     * @brief This function handles TIM2 global interrupt.
202     */
203 void TIM2_IRQHandler(void)
204 {
205     /* USER CODE BEGIN TIM2_IRQHandler 0 */
206
207     /* USER CODE END TIM2_IRQHandler 0 */
208     HAL_TIM_IRQHandler(&htim2);
209     /* USER CODE BEGIN */
210
211     /* USER CODE END */
212 }
213
214 /* USER CODE BEGIN */
215
216 /* USER CODE END 1
217 ****
218

```

A context menu is open at the line `HAL_TIM_IRQHandler(&htim2);`. The menu items include:

- Split Window horizontally
- Insert '#include file'
- Go to Headerfile "stm32f1xx_it.h"
- Insert/Remove Breakpoint** (radio button selected)
- Enable/Disable Breakpoint
- Go To Definition Of 'HAL_TIM_IRQHandler(htim2);'
- Go To Reference To 'HAL_TIM_IRQHandler(htim2);'
- Insert/Remove Bookmark
- Undo
- Redo
- Cut
- Copy
- Paste
- Select All
- Outlining
- Advanced

The status bar at the bottom right shows: electronics *****END OF FILE***** L:208

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 58

Trong phần Define hàm đó các bạn sẽ tìm thấy hàm Call_back cho ngắt Input Capture, Go to Define hàm đó copy và paste vào phần tiền xử lý trên hàm main()

The screenshot shows a code editor with the following code snippet:

```

3165     */
3166 void HAL_TIM_IRQHandler
3167 {
3168     /* Capture compare 1
3169     if (__HAL_TIM_GET_FLAG)
3170     {
3171         if (__HAL_TIM_GET_IT)
3172         {
3173             {
3174                 HAL_TIM_CLEAR_FLAG(htim->Channel =
3175
3176                 /* Input capture
3177                 if ((htim->Instance == 0)
3178                 {
3179 #if (USE_HAL_TIM_REGISTER_CALLBACKS)
3180                     htim->IC_Capt
3181 #else
3182                     HAL_TIM_IC_Ca
3183 #endif /* USE_HAL_TIM_REGISTER_CALLBACKS */
3184
3185                 }
3186                 /* Output compare event */

```

A context menu is open at the line `HAL_TIM_IC_CaptureCallback`. The menu items include:

- Split Window horizontally
- Insert '#include file'
- Go to Headerfile "stm32f1xx_hal_tim.h"
- Insert/Remove Breakpoint** (radio button selected)
- Enable/Disable Breakpoint
- Go To Definition Of 'HAL_TIM_IC_CaptureCallback'
- Go To Reference To 'HAL_TIM_IC_CaptureCallback'
- Insert/Remove Bookmark
- Undo
- Redo
- Cut
- Copy
- Paste
- Select All
- Outlining
- Advanced

The status bar at the bottom right shows: != 0x00U

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 59

Trong main.c chúng ta tạo một biến chứa Giá trị đọc được từ Counter, trong hàm xử lý ngắt ta sẽ phân luồng ngắt, đọc giá trị Capture, sau đó xóa giá trị Counter về 0.

```

61 /* Private user code -----*/
62 /* USER CODE BEGIN 0 */
63 void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
64 {
65     /* Prevent unused argument(s) compilation warning */
66     UNUSED(htim);
67     if(htim->Instance == TIM2) // Ngắt IC thuộc TIM2
68     {
69         u16_CaptureVal = __HAL_TIM_GET_COMPARE(&htim2,TIM_CHANNEL_1); // Đọc giá trị Capture
70         __HAL_TIM_SET_COUNTER(&htim2,0); // Xóa Counter
71     }
72     /* NOTE : This function should not be modified, when the callback is needed,
73             the HAL_TIM_IC_CaptureCallback could be implemented in the user file
74     */
75 }
76 /* USER CODE END 0 */
77

```

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 60

Trước While(1), chúng ta viết các hàm khởi chạy IC và OC

```

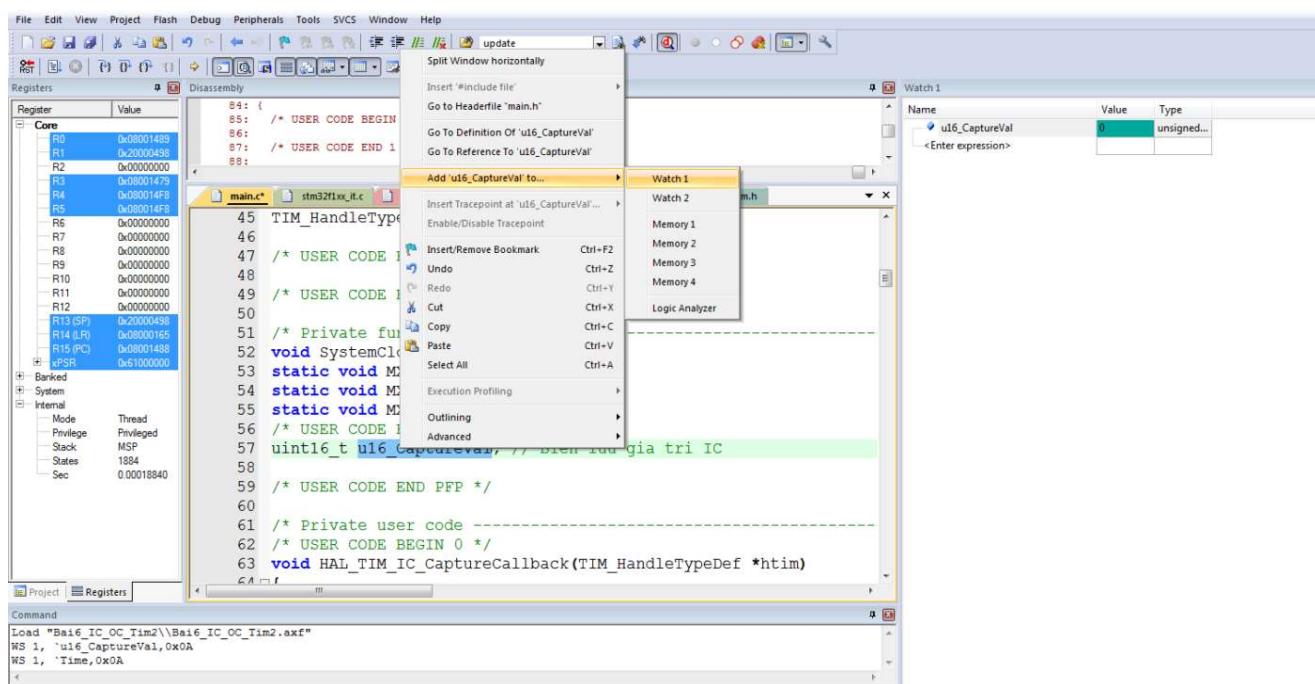
103 /* Initialize all configured peripherals */
104 MX_GPIO_Init();
105 MX_TIM2_Init();
106 MX_TIM3_Init();
107 /* USER CODE BEGIN 2 */
108 HAL_TIM_IC_Start_IT(&htim2, TIM_CHANNEL_1); // Bật IC chế độ Ngắt
109 HAL_TIM_OC_Start(&htim3,TIM_CHANNEL_1); // Bật OC không ngắt
110 /* USER CODE END 2 */
111
112 /* Infinite loop */
113 /* USER CODE BEGIN WHILE */
114 while (1)
115 {
116     /* USER CODE END WHILE */
117
118     /* USER CODE BEGIN 3 */
119
120 }
121 /* USER CODE END 3 */
122

```

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 61

Build và nạp chương trình vào Kit.

Kết nối chân PA0 và chân PA6 vào nhau. (Timer 2 sẽ đọc xung từ Timer 3 truyền tới). Bật Debug, click vào tên của biến u16_CaptureVal chọn Add to Watch 1



Bài 6: STM32 Timer chế độ Input Capture và Output Compare 62

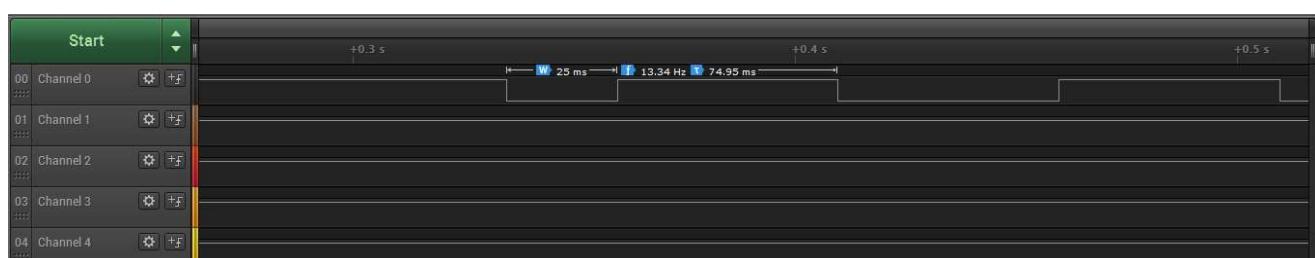
Nhấn Run(F5) sẽ thấy u16_CaptureVal có giá trị là 100

Name	Value	Type
u16_CaptureVal	100	unsigned...
<Enter expression>		

Bài 6: STM32 Timer chế độ Input Capture và Output Compare 63

Để giải thích vì sao có giá trị đó các bạn dùng bộ Logic Analyzer Cắm vào CH1 của Timer 3 để đọc xung của nó.

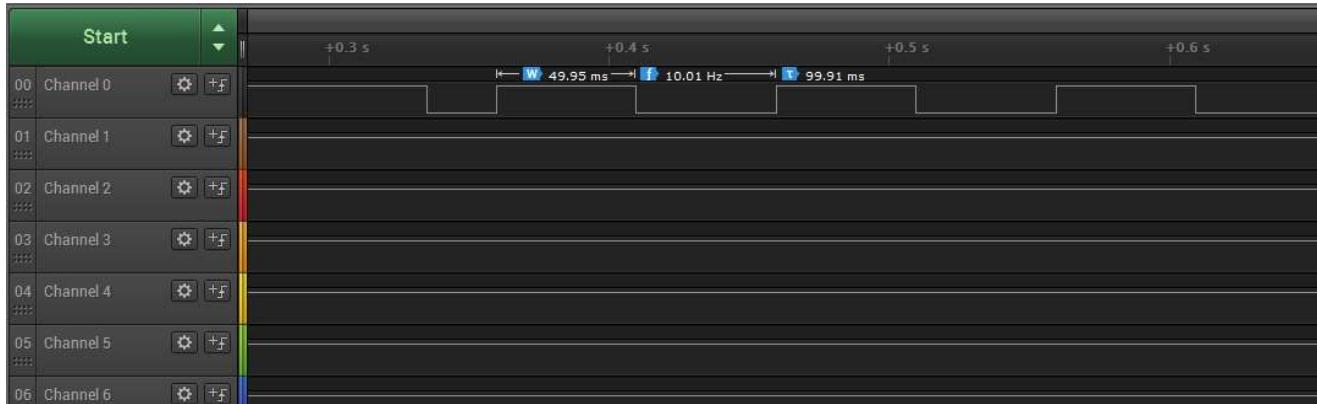
Nếu chưa biết sử dụng bạn đọc bài viết: Hướng dẫn sử dụng Logic Analyzer



Bài 6: STM32 Timer chế độ Input Capture và Output Compare 64

Khi bắt đầu chạy Timer, kênh CH1 sẽ đảo trạng thái đó là khoảng 25ms đầu tiên, sau đó Timer 3 đếm đến 50ms rồi quay về 0, đến 25ms tiếp theo lại đảo trạng thái.

Vì vậy mỗi lần có xung lên sẽ cách nhau 2 lần đảo trạng thái. Nghĩa là 100ms.



Bài 6: STM32 Timer chế độ Input Capture và Output Compare 65

Vì vậy giá trị Capture được sẽ có giá trị là 100.

Kết

Vậy là chúng ta đã học được cách sử dụng STM32 Timer chế độ Input Capture và Outout Compare.

Đừng quên chia sẻ nếu thấy bài viết này có ích nhé

5/5 - (5 bình chọn)

Related Posts:

1. [Bài 12: Lập trình STM32 với giao thức SPI](#)
2. [Bài 11: Lập trình STM32 với Giao thức UART](#)
3. [Bài 7: STM32 Timer chế độ PWM](#)
4. [Bài 4: Lập trình Ngắt Ngoài STM32 EXTI](#)
5. [Bài 3: Lập trình STM32 GPIO điều khiển Led và nút nhấn](#)
6. [Bài 2: Tổng quan về KIT STM32F103C8T6 Blue Pill](#)



KHUÊ NGUYỄN

Chỉ là người đam mê điện tử và lập trình. Làm được gì thì viết cho anh em xem thôi. :D

10 THOUGHTS ON “BÀI 6: STM32 TIMER CHẾ ĐỘ INPUT CAPTURE VÀ OUTPUT COMPARE”



minh anh says:

ad cho mình hỏi, tại sao lại kết nối chân PA0 và PA6 vào nhau ?

03/03/2021 AT 2:10 SÁNG

TRẢ LỜI



minh anh says:

à mình hiểu r PA0 là TIME2, PA6 là Timer 3

03/03/2021 AT 2:18 SÁNG

TRẢ LỜI



Khuê Nguyễn says:

Đúng rồi, mục đích là 1 timer đọc tín hiệu, 1 timer phát tín hiệu

03/03/2021 AT 12:35 CHIỀU

TRẢ LỜI



minh anh says:

Cảm ơn ad nhiều, ad cho mình hỏi thêm muốn mô phỏng trên proteus cần set chân PA0 và chân PA6 không, vì mình mô phỏng k thấy tín hiệu ở 2 chân đó.

Thanks ad nhiều!



Khuê Nguyễn says:

m ko dùng mô phỏng, cũng khuyên bạn ko nên dùng mô phỏng, con bluepill rẻ lắm bạn mua về dùng nó trực quan với đúng hơn

03/03/2021 AT 11:03 CHIỀU

TRẢ LỜI



Phuoc Thien DO says:

ad hướng dẫn cách đo độ rộng xung luôn đi ad

05/03/2021 AT 11:07 CHIỀU

TRẢ LỜI

Nguyễn Hoàng Danh says:

```
/* Channel 1, Configuration in OC mode */
TIM_OCStructInit(&TIM5_OCIInitStructure);
TIM5_OCIInitStructure.TIM_OCMode = TIM_OCMode_Timing;
TIM5_OCIInitStructure.TIM_OutputState = TIM_OutputState_Disable;
TIM5_OCIInitStructure.TIM_OutputNState = TIM_OutputNState_Disable;
TIM_OC1Init(TIM5, &TIM5_OCIInitStructure);
TIM_OC1PreloadConfig(TIM5, TIM_OCPreload_Disable);
Em đang gấp dính TIM5 xài m mode timing anh có thể giải thích cho em cái
state với nstate k anh
```

16/07/2021 AT 12:51 SÁNG

TRẢ LỜI

Foli says:

Anh nên chú thích các thanh nữa nhe.. em đọc vô chả hình dung được hết
^^!.

-
- Counter Register (TIMx_CNT): Khi hoạt động, thanh ghi này tăng hoặc giảm giá trị theo mỗi xung clock đầu vào. Tùy vào bộ timer mà counter này có thể là 16bit hoặc 32bit.
 - Prescaler Register (TIMx_PSC): Giá trị của thanh ghi bộ chia tần (16bit)

cho phép người dùng cấu hình chia tần số đầu vào (CK_PSC) cho bất kì giá trị nào từ [1- 65536]. Sử dụng kết hợp bộ chia tần của timer và của RCC giúp chúng ta có thể thay đổi được thời gian của mỗi lần CNT thực hiện đếm, giúp tạo ra được những khoảng thời gian, điều chế được độ rộng xung phù hợp với nhu cầu.

– Auto-Reload Register (TIMx_ARR): Giá trị của ARR được người dùng xác định sẵn khi cài đặt bộ timer, làm cơ sở cho CNT thực hiện nạp lại giá trị đếm mỗi khi tràn (overflow khi đếm lên – CNT vượt quá giá trị ARR, underflow khi đếm xuống – CNT bé hơn 0). Tùy vào bộ timer mà counter này có thể là 16bit hoặc 32bit.

04/12/2021 AT 7:30 SÁNG

TRẢ LỜI

Khuê Nguyễn says:

Cám ơn e góp ý, a sẽ thêm nhé

04/12/2021 AT 3:12 CHIỀU

TRẢ LỜI

Foli says:

Vâng ^^.

04/12/2021 AT 5:35 CHIỀU

TRẢ LỜI

Trả lời

Email của bạn sẽ không được hiển thị công khai. Các trường bắt buộc được đánh dấu *

Bình luận *

Tên *

Email *

Trang web

PHẢN HỒI

Fanpage

Khuê Nguyễn Creator - Học Lập Trình Vi Điều Khiển
2.754 lượt thích

Đã thích Chia sẻ

Khuê Nguyễn Creator - Học Lập Trình Vi Điều Khiển
khoảng một tháng trước

Lý do thời gian gần đây mình không viết bài và làm thêm gì cả là đây 😊)
Chính thức ra mắt sản phẩm định vị thông minh vTag.

Đây là một sản phẩm định vị đa năng với 3 công nghệ định vị WIFI, GPS, LBS kết hợp với sóng NB-IOT dành riêng cho các sản phẩm IOT.

Chỉ với 990.000đ chúng ta đã có thể có sản phẩm để:

- Định vị trẻ em, con cái... [Xem thêm](#)

Bài viết khác

Lập trình 8051 - AT89S52



Khuê Nguyễn Creator



Bài 1: Tổng quan về 8051 và chip AT89S51 - 52



Tổng quan về 8051

8051 là một dòng chip nhập môn cho lập trình viên nhúng, chúng được sử...

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator



Lập trình STM32 HID Host giao tiếp với chuột và bàn phím

Lập trình STM32 USB HID Host giao tiếp với chuột và bàn phím máy tính

Trong bài này chung ta sẽ cùng học STM32 HID Host, biến STM32 giống như...

[ĐỌC THÊM](#)



Lộ trình học lập trình nhúng từ A tới Z

Lập trình nhúng là một ngành có cơ hội nhưng cũng đòi hỏi nhiều kiến...

3 COMMENTS

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator





Lập trình STM32F407 SDIO đọc dữ liệu thẻ nhớ

Lập trình STM32 SDIO đọc ghi dữ liệu vào thẻ nhớ SD card

Trong bài này chúng ta cùng học cách lập trình STM32 SDIO, một chuẩn giao...

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator



Lập trình STM32F407 DAC chuyển đổi số sang tương tự

Lập trình STM32 DAC tạo sóng hình Sin trên KIT STM32F407 Discovery

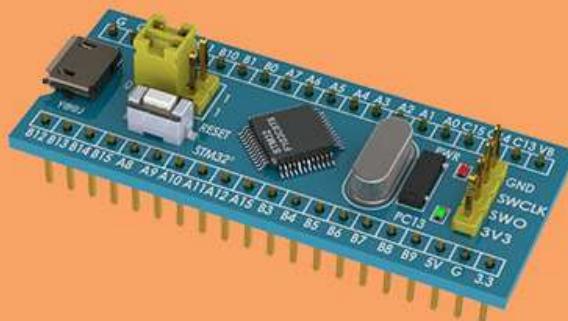
Trong bài này chúng ta sẽ cùng nhau tìm hiểu STM32 DAC với KIT STM32F407VE...

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator



Sử dụng hàm printf để in Log khi Debug trên STM32

Hướng dẫn sử dụng printf với STM32 Uart để in Log trên Keil C

Trong bài này chúng ta sẽ học cách retarget hàm printf của thư viện stdio...

3 COMMENTS

[ĐỌC THÊM](#)

ESP32 và Platform IO



Khuê Nguyễn Creator



Bài 9 WIFI: Lập trình ESP32 OTA nạp firmware trên Internet

Lập trình ESP32 FOTA nạp firmware qua mạng Internet với OTA Drive

Trong bài này chúng ta sẽ học cách sử dụng ESP32 FOTA (Firmware Over The...)

4 COMMENTS

[ĐỌC THÊM](#)

Lập trình Nuvoton



Khuê Nguyễn Creator



Cài đặt SDC Complier và Code:Blocks IDE

Hướng dẫn cài đặt SDCC và Code::Blocks lập trình Nuvoton

Ở bài này chúng ta sẽ cài đặt các công cụ cần thiết cho việc...

[ĐỌC THÊM](#)



Blog này làm ra để lưu trữ tất cả những kiến thức, những câu chuyện của mình. Đôi khi là những ý tưởng nhất thời, đôi khi là các dự án tự mình làm. Chia sẻ cho người khác cũng là niềm vui của mình, kiến thức mỗi người là khác nhau, không hẳn quá cao siêu nhưng sẽ có lúc hữu dụng.

Liên Kết

Nhóm: Nghiên Lập Trình

Fanpage: Khuê Nguyên Creator

My Shop

Thông Tin

Tác Giả

Chính Sách Bảo Mật



Copyright 2022 © Khuê Nguyễn