

T.O.M.A.S Team







- Practice a bit with STMStudio monitoring variables and creating expressions
- Practice a bit with printf implementation using SWO channel and STLink Utility application
- Practice a bit with printf implementation using USART and any terminal application





In our next task we will use a PC applications to monitor data sent by Nucleo board over STLink v2.1:

- asynchronously using USART2 interface -> monitor on any terminal application
- synchronously, using SWO interface -> monitor on STLink Utility application





Complementary debug tools Using printf() over SWO and USART2 to send data via USB to PC



- On most of STM32 (except STM32F0, L0 devices) there is peripheral called Instrumentation Trace Macrocell (ITM) (do not mix with Enhanced Trace Macrocell - ETM) available
- This peripheral can be used to send-out data from MCU over Single Wire Output (SWO) pin
- It is possible to redirect **printf()** to use this peripheral
- Most IDEs can display this information during debug
- On 64pins Nucleo boards (NUCLEO_L476RG as well) SWO pin (PB3) is connected to the STLink (SB15 solder bridge is closed) so no additional connection is required to reuse this line

Using SWO introduction







Concept of the system

sending measured data over SWO and USART2 via STLink

- Use the project from previous part (L4_DAC_ADC.ioc)
- Enable SWO line with debug interface (PB3 pin)
- Enable USART2 in asynchronous mode: 115200bps, 8bit data, no parity, 1bit stop
- Modify printf() to use SWO and USART2 as an output channels
- Configure user button (PC13) to start/stop Timer2
- Configure ADC transfer complete callback to send adcbuf[0] using printf







Adding SWO, EXTI13 and USART2 STM32CubeMX – Pinout tab

- 1. Open L4_DAC_ADC.ioc project in STM32CubeMX
 - Menu File → Open Project
- 2. In STM32CubeMX, pinout tab enable SWO pin
 - Select SYS → Debug → Trace Asynchronous Sw



3. Click left button on mouse over PC13 pin and select GPIO_EXTI13 mode



- 4. Select USART2 in asynchronous mode
 - Select USART2->Mode: Asynchronous





Configure USART2 and EXTI13 STM32CubeMX – Configuration tab 8

- Configure USART2 parameters:
 - 115200 bps / 8 bit data / 1 bit stop / no parity / no HW control

USART2 Configuration	USARTZ
Parameter Settings 🏼 🚽 Us	er Constants 🛷 NVIC Settings 🗹 DMA Settin
Configure the below parameter	5:
Search : Search (Crtl+F)	👻 🛧
🖃 Basic Parameters	
Baud Rate	115200
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1
Advanced Parameters	
Data Direction	Receive and Transmit
Over Sampling	16 Samples
Single Sample	Disable
Advanced Features	
Auto Baudrate	Disable
TX Pin Active Level 1	nversion Disable
RX Pin Active Level 1	nversion Disable
Data Inversion	Disable
TX and RX Pins Swa	pping Disable
Overrun	Enable
DMA on RX Error	Enable
MSB First	Disable

Enable EXTI line[15:10] within NVIC configuration

NVIC Conf	iguration	WVIC	·			X
VVIC 🕑 🕻	Code generation					
Priority Group	4 bits for pre-emption p	riority 0 bits f 💌	Sort by F	Premption Priority a	ind Sub Pro	rity
Search	Search (Crtl+F)		Show or	nly enabled interru	pts	
Interrupt Table	•		Enab	Preemption Pri	Sub Pri	
Non maskable i	nterrupt		1	0	0	
Hard fault inter	rupt		1	0	0	
Memory manag	ement fault		1	0	0	
Prefetch fault, r	nemory access fault		1	0	0	
Undefined instru	uction or illegal state		1	0	0	
System service	call via SWI instruction		1	0	0	
Debug monitor			1	0	0	
Pendable reque	st for system service		1	0	0	
Time base: Sys	tem tick timer		1	0	0	
PVD/PVM1/PVM	2/PVM3/PVM4 interrupts	through EXTI lines 16/		0	0	
Flash global inte	errupt			0	0	
RCC global inte	rrupt			0	0	
DMA1 channel1	global interrupt		1	0	0	
DMA1 channel3	global interrupt		1	0	0	
ADC1 and ADC2	2 interrupts			0	0	
TIM2 global inte	errupt			0	0	
USART2 global	interrupt			0	0	
EXTI line[15:10] interrupts		V	0	0	
T ING global Inte	errupt, DAC channel1 and	channelz underrun err		U	U	
FPU global inter	rupt			0	0	



- Generate the code with added new features
- Perform further processing in SW4STM32 (L4_DAC_ADC project)







Using SWO for printf in gcc

In main.c source file:

- include the stdio.h library to make printf working
- define _write() function used to send data over SWO using ITM_SendChar() function
- as ITM_SendChar() function is accepting single character, we should send data character by character in the loop.
- add some messages at the beginning of the application
- compile and run the code

```
/* USER CODE BEGIN Includes */
#include <stdio.h>
/* USER CODE END Includes */
```

```
/* USER CODE BEGIN 4 */
```

```
int _write(int file, char *ptr, int len)
```

```
int DataIdx;
```

}

```
for(DataIdx=0; DataIdx<len; DataIdx++)</pre>
```

```
ITM SendChar(*ptr++);
```

```
/* USER CODE END 4 */
```

```
printf("Application start.\n");
printf("Press User button to start new acquisition\n");
/* USER CODE END 2 */
```





Using USART2 for printf in gcc

In main.c source file:

- include the stdio.h library to make printf working
- define _write() function used to send data over USART2
- add some messages at the beginning of the application

```
/* USER CODE BEGIN Includes */
#include <stdio.h>
/* USER CODE END Includes */
```

```
/* USER CODE BEGIN 4 */
```

```
int _write(int file, char *ptr, int len)
```

```
HAL_UART_Transmit(&huart2,ptr,len,10);
return len;
```

```
/* USER CODE END 4 */
```

```
printf("Application start.\n");
printf("Press User button to start new acquisition\n");
```

```
/* USER CODE END 2 */
```





Sending ADC results over printf

```
• In main.c source file (inside USER CODE
volatile uint8 t flag=1;
                                                     section) implement own EXTI13 and ADC
/* USER CODE END PV */
                                                     conversion complete callbacks:
/* USER CODE BEGIN 4 */
void HAL GPIO EXTI Callback(uint16 t GPIO Pin)
                                                         First one to control start/stop acquisition
                                                         Second one for sending adcbuf[] using printf() once
 if (GPIO PIN 13==GPIO Pin)
                                                         acquisition is stopped
   if(0==flag)
     HAL TIM OC Stop(&htim2, TIM CHANNEL 2);
     printf("Acquisition stopped\n");
                                                     void HAL ADC ConvCpltCallback(ADC HandleTypeDef* hadc)
     printf("Press button to START a new one\n");
     flag=1;
                                                      printf("%d\n",adcbuf[0]);
    else
                                                     /* USER CODE END 4 */
     printf("Acquisition started\n");
     printf("Press button to STOP it\n");
     flag=0;
                                                       Additionally we should not start Timer2 at the
     HAL TIM OC Start(&htim2,TIM CHANNEL 2);
                                                       beginning (USER CODE 2 section, before
                                                       while(1) loop)
```

Analize the data from ADC

- Open serial terminal selecting COM port assigned to Virtual COMP Port (VCP) implemented in STLink, using the configuration:
 - 115200pbs / 8 data bits / 1 STOP bit / NO parity / NO HW flow control
- Reset the board, now you should see "Application start" message in terminal window
- Follow the instructions in the terminal window (User button is the blue one on Nucleo board)
- Using copy&paste mechanism copy data to clipboard and then paste them into the spread sheet application (i.e. Excel)
- Display the waveform



Chart Title



COM85 115200 bps, 8N1, no handshake Settings Clear About 9 Application start. Press User button to start new acquisition Acquisition started Acquisition started Acquisition stopped Please wait for the data 20	Close
Application start. Press User button to start new acquisition Acquisition started Acquisition stopped Please wait for the data	_
Press User button to start new acquisition Acquisition started Acquisition stopped Please wait for the data	
Acquisition started Acquisition stopped Please wait for the data	
Acquisition stopped Please wait for the data	
Please wait for the data	
170	
18	
2049	
2822	
3491	
3933	
4026	
3946	
3473	
2829	
1050	
1202 560	
100	
86	
126	
587	
1234	
2038	
2820	
3492	
3942	
4026	
3941	
3484	
2824	
2027	
1248	
584	
122	
84	
128	
584	
Press User button to start new acquisition	

Analize the data from ADC

Open STLink Utility

- Connect to the board (*Target → Connect*)
- Open SWO viewer window (ST-LINK → Printf via SWO viewer)
- Update System clock to 80 000 000 and Stimulus to 0 (toolchain settings)
- Start data catching → Start button
- Grab the ADC data
- Using copy&paste mechanism copy data to clipboard and then paste them into the spread sheet application (i.e. Excel)
- Display the waveform

Hint: in case there is nothing in trace window, please check the version of STLink firmware -> see the next slide



Serial Wire Viewer Settinas STOP 180000000 System clock Printf data number: 1233 Application start Press User button to start acquisition Acquisition started Acquisition stopped Please wait for the data ... 182 2057 2842 3509 3951 4038 3951 3505 2842 2056 1270 606 216 181 216 607 1270 2056 2841 3507 3950 4037 3950 3506 2842 2056 1269 607 216 182 216 607 Press User button to start new acquisition Clear

data sent over SWO

SWO viewer in STLink Utility

No data in trace window – upgrade of the driver

- In case there is no data in trace window, it is highly probable that the STLink software on Nucleo board is not up-to-date
- To update this software, please follow the below procedure:
 - Select ST-LINK->Firmware update
 - Press Device Connect button on "ST-Link Upgrade" window
 - After a while there will be information about firmware version on STLink and the current one
 - In case the current one has higher number, please upgrade STLink by pressing Yes >>>> button
 - After completion of the operation message window will appear
 - We can close "ST-Link Upgrade" window and continue operations on upgraded board.





What have we learnt 16

- Practice a bit with STMStudio monitoring variables and creating expressions
- Practice a bit with printf implementation using SWO channel and STLink
 Utility application
- Practice a bit with printf implementation using USART and any terminal application







www.st.com/mcu

