Современные 32-разрядные ARM-микроконтроллеры серии STM32: среда разработки программ СооСох CoIDE

Олег Вальпа (г. Миасс, Челябинская обл.)

В статье приведено описание программного инструмента CooCox CoIDE, предназначенного для разработки и отладки программного кода на языке Си для 32-разрядных ARM-микроконтроллеров серии STM32 от компании STMicroelectronics.

Введение

В настоящее время в мире существует множество программных сред разработки для микроконтроллеров серии STM32. Одним из таких продуктов является среда разработки CooCox CoIDE, основанная на Eclipse.

CooCox CoIDE						
Eile Edit ⊻iew Project Elash Debug	; Search Help					
📫 🗎 🔄) 🏙 🏙 🍄 🤐 🏛 🖉 醌] .	<mark>∻ →</mark>] & → 원 → ♥⇒ ↔ → →					
Device [/]	Repository 🛛 🗖 🗖	🕼 Help 🛛 🕃 Outline 📄				
	Step 1 Select Manufacturer					
	ARM					
	Atmel					
	Energy Micro					
	ц					
	NXP					
	Nuvoton					
	SI					
î p1						
	4					
	Manufacturers Chips Components					
Sign in to CooCox						





Рис. 2. Стартовое окно Step 2 – характеристики микроконтроллера

В отличие от Keil, IAR и других сред разработки, CooCox CoIDE проста в освоении и не стоит денег. Кроме того, она содержит множество библиотек и встроенный отладчик ST-Link, что позволяет осуществить быстрый старт.

СооСох СоIDE является средой разработки для микропроцессорных устройств и базируется на Eclipse. Кроме серии STM32 она поддерживает множество других семейств микроконтроллеров, таких как Freescale, Holtek, NXP, Nuvoton, TI, Atmel SAM, Energy Micro и другие. С развитием продукта CoIDE список поддерживаемых микроконтроллеров постоянно расширяется.

Описание среды разработки

Рассмотрим процедуру установки среды на персональном компьютере, для чего создадим простую программу. В качестве микроконтроллерного устройства будем использовать широко распространённую отладочную плату STM32VLDiscovery [1] от фирмы STMicroelectronics с установленным на ней микроконтроллером STM32F100RBT6B.

Для установки описываемой среды разработки необходимо загрузить на персональный компьютер последнюю версию инсталляционного файла с официального сайта СооСох [2]. Перед загрузкой потребуется пройти простую и бесплатную процедуру регистрации на сайте производителя продукта. После чего следует инсталлировать загруженный файл на компьютере. После успешной установки среды CoIDE необходимо запустить её. При этом на экране монитора компьютера должно появиться стартовое окно Step 1, представленное на рисунке 1, в котором необходимо выбрать производителя микроконтроллера.

Для нашего случая это будет производитель ST. При его выборе будет автоматически выполнен переход к следующему шагу Step 2 по выбору микроконтроллера. Здесь следует выбрать тип микроконтроллера STM32F100RB. При этом в правой части окна отображаются краткие характеристики конкретного микроконтроллера, как показано на рисунке 2.

После выбора микроконтроллера автоматически произойдёт переход к третьему шагу Step 3 (см. рис. 3) с целью выбора необходимых для работы библиотек.

Для примера создадим простейший проект, осуществляющий мигание светодиодом. Для этого нам понадобится библиотека GPIO. При её подключении, путём установки флажка, CoIDE попросит создать новый проект. В ответ на этот запрос требуется нажать программную кнопку Yes, указать папку, где будет храниться проект, и присвоить ему название, например, как показано на рисунке 4.

После этого CoIDE подключит к проекту другие необходимые для работы библиотеки и создаст всю необходимую структуру проекта. Во вкладке Help можно увидеть, что для каждой выделяемой библиотеки отобража-

le <u>E</u> dit ⊻iew Project Elash <u>D</u>	ebug Se <u>a</u> rch <u>H</u> elp					
1 🗋 😨 🛗 🏙 🏙 😵 🖉 🗄 🗄	毘│┩╾│カ ァ ∛ ァ や ← ァ →	-				
evice [STM32F100RB]	Repository 🛛	- 0	lelp X			
Common	Step 3 Select Components	[ST/STM32F100RB	GPIO			
Peripheral.ST			Overview			
• RCC (with 1 example)	C Library	Implement t	Each Port have 16 pins, e	ach I/O port bit is		
• GPIO (with 4 examples)	🔲 🔲 Retarget printf	Implementat	freely programmable to modes as follows: Input			
Boot	CMSIS core	CMSIS core t	Input, Output open-drain, Output push-pull,			
└ ● CMSIS Boot	BOOT		function open-drain. All G	ull and Alternate PIOs are APB2		
	CMSIS Boot	STM32F10x	peripheral. In output mod modify only one or severa	e, it is possible to al bits in a single		
	B PERIPHERAL ST		atomic APB2 write access. This component can be used to set pins mode and speed, to input and output, to set pins as Event output or EXTI Line, to remap the pin, to selects the Ethernet media interface, tp lock GPIO Pins configuration.			
	🛓 🗹 RCC	STM32F10x				
	CRC	STM32F10x				
	BKP	STM32F10x				
	🔤 🎽 🗹 GPIO	STM32F10x	API Reference			
🖆 p1	EXTI	STM32F10x	GPIO DeInit	Deinitializes t		
	DMA	STM32F10x		GPIOx peripheral		
	DAC	STM32F10x		registers to their default		
	RTC	STM32F10x		reset values.		
	IWDG	STM32F10x	X GPIO AFIODeInit	Deinitializes t		
	WWDG	STM32F10x		Functions Functions (remap, event control and EXTI configuration) registers to their default reset values.		
	SPI	STM32F10x				
	🛓 🗖 12C	STM32F10x				
		STM32F10x				
	FLASH	STM32F10x	1			
			GPIO Init	Initializes the GPIOx peripheral		
	Manufacturara China Compor	ponte		according to t		

Рис. 3. Стартовое окно Step 3 - выбор библиотек

ются примеры её применения. Кроме того, во вкладке Device отображается несколько готовых примеров. CoIDE позволяет загружать эти примеры из среды разработки непосредственно в проект. В дальнейшем данную группу можно будет пополнять своими примерами.

🚺 New Project		1 Debug Configurations	×
Project Create a new project resource.		Create, manage, and run configurations Ready to launch	Ť.
Project name: p1 Use default location Location: B:\STM32PRJ	Browse	Image: Structure Image: Structure Imag	▼ Port SWL ▼
Einish	Cancel		Close

Рис. 4. Окно создания нового проекта

```
Листинг
#include «stm32f10x.h»
#include «stm32f10x_gpio.h»
#include «stm32f10x_rcc.h»
int main(void)
ſ
    int i:
    /* Инициализация индикаторов на плате STM32 */
    GPI0_InitTypeDef GPI0_InitStructure;
    /* Инициализировать индикаторы, подключенные к выводам РС6,9 и
включение синхросигнала*/
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC, ENABLE);
    /* Конфигурирование выводов GPIO */
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_6|GPIO_Pin_9;
    GPI0_InitStructure.GPI0_Mode = GPI0_Mode_Out_PP;
    GPIO InitStructure.GPIO Speed = GPIO Speed 50MHz;
    GPIO_Init(GPIOC, &GPIO_InitStructure);
    while (1)
    {
    /* Переключение индикатора подключенного к выводу PC6 */
    GPIOC->ODR ^= GPIO_Pin_6;
    /* Задержка */
    for(i=0;i<0x100000;i++);</pre>
    /* Переключение индикатора подключенного к выводу PC9 */
    GPIOC->ODR ^= GPIO_Pin_9;
    /* Задержка */
    for(i=0;i<0x100000;i++);</pre>
    }
}
```

На рисунке 3 видно, что в примерах уже присутствует программный код с названием GPIO_Blink для мигания светодиодом. Если нажать в строке этого примера программную кнопку Add, то этот код добавится в проект как подключаемый файл. Но можно поступить и по-другому. Для этого просто откройте пример с помощью программной кнопки View, выделите весь программный код и скопируйте его в буфер обмена, а затем откройте файл проекта с именем main.c и вставьте в него скопированный код, предварительно удалив все имеющиеся в нём строки. После чего следует заменить строку void GPIO_Blink(void) на int main(void). Таким образом, мы получим проект с главным файлом main.c, содержащим основную функцию main. Программный код данного файла, с переведёнными на русский язык комментариями, представлен в листинге.

Теперь необходимо выполнить компиляцию проекта. Поскольку среда использует для компиляции проекта распространяемый бесплатно компилятор GCC, его следует установить на компьютер. Для этого необходимо открыть сайт GNU Tools for ARM Embedded Processors [3] и выбрать в правой части страницы инсталляционный файл компилятора для операционной системы, которая установлена на персональном компьютере. После его загрузки следует инсталлировать данный файл на компьютер. Теперь можно скомпилировать проект путём нажатия клавиши F7 на клавиатуре или выбрать в меню Project-> Build.

Рис. 5. Окно загрузки программного кода в микроконтроллер

После успешной компиляции проекта необходимо загрузить полученный программный код в микроконтроллер. Для этого необходимо подключить отладочную плату к компьютеру через интерфейс USB. В настройках Debug Configuration необходимо выбрать отладчик ST-Link, как это показано на рисунке 5.

Для загрузки программы в микроконтроллер необходимо в главном меню среды выбрать Flash->Program Download или кликнуть левой кнопкой мыши по соответствующей иконке на панели инструментов.

После окончания загрузки на отладочной плате начнёт мигать светодиод.

При необходимости в среде CoIDE можно использовать встроенный отладчик, который активируется путём нажатия клавиш Ctrl+F5 или в главном меню Debug->Debug.

Литература

- 1. www.st.com.
- 2. www.coocox.org/software/coide.php.
- 3. www.launchpad.net/gcc-arm-embedded.