

Introduction to the STM32CubeIDE for STM32 Microcontrollers

workspace_1.3.0 - Device Configuration File Edit Source Refactor Navigate S	n Tool - STM32CubeIDE iearch Project Run Window	Help														-	6	×
📑 • 🗟 🐘 🖲 • 🗞 • 🛍 🔪 💩	12 • 63 • 12 • 69 • 14	• O • 💁 • 😕	A • 1 50 1	표 [요] = 종[-	• 🗇 🔶 • 🔿 •	- 🗗 0									Q	uick Access	8	
Project Explorer 🐹 👘 🗖	🛄 Blink.ioc 🕄 💽 main.c	🕞 main.h 🛛 🖻 str	n32f1xx_hal_con	f.h														8
📄 🔩 🔍	Pinout (Clock Configuration					Project	Manager				Tools			8
> 🗱 Binaries					Add	ditional So	ftware	~	Pinout									Ĩ
> 🔊 Includes 🗸 🔁 Core	Q Ø			GPIO Mod	le and Configuration	ion						😂 Pinout vie	w System	m view				
V 😂 Inc	Categories A->Z			C	onfiguration				4									1
> h stm32f1xx_hal_conf.h	System Core 🗸 🗸	Group By Periphe	rals					~										
> h stm32f1xx_it.h	÷	😔 GPIO 🛛 📀 R	ICC															
> 🗁 Startup	DMA GPIO	Coardo Signalo																
> 🗁 Univers > 👝 Debug	IWDG	Search /Crt/+P	-				Show only Mo	dified Pins						-				
Blink.ioc	✓ RCC	Pin Name + Size	al on Pin GPIC	outo GPIO m	oda GRIO Pull.	Maximum	Liser Label	Modified	d - 1			9 9 8 8	88 86 86	ANE MON				
> III Test	VWDG	PC13-TAM n/a	Low	Output P	us No pull-up	. High	LED				LED			V00				
												1016. 1015- ca		PA13 PA12				
	ânaloa 🔾										RCC_OSC_N			PATT				
	Printing										100,000,001	MST		PAD				
	Timers >											/00A STM	32F103C8Tx	PINS PILIS				
	Connectivity >	PC13-TAMPER-R	TC Configuratio	n :							P816 P813							
	Computing >	GPIO output level		Low			Low ~				89 99							
	Middleware >	GPIO mode			Output Push P													
		GPIO Pull-up/Pul	-down		No pull-up and no pull-down		~											
		Maximum output	speed		High													
		User Label			LED					Q []) Q				Q	~		
	Problems 😂 🎜 Tasks 🖂	Console 🔲 Propertie	5				<u>∌</u> ⊽ 1	1	Build Analyze	r 23 al Static Stat	ck Analyzer					et.	~ ~ 0	
	0 items								Blink.elf - /Bli	ink/Debug - May	1, 2020 5:18:48	AM						
	Description		Resource	Path	Location	Type			Memory Regio	ns Memory Details								
									Region	Start address	End address	Size	Free	Used	Usage (S	6		
									E FLASH	0x20000000 0x08000000	0x20005000 0x08010000	20 KB 64 KB	18.45 KB 59.55 KB	1.55 KB 4.45 KB	7.73% 6.95%			
E P Type here to search	0	it 📒	()	U	🥶 📧	0 👤									~	U (1 4) 6	22 AM	20
																2	172020	9

DOWNLOAD PRINTER-FRIENDLY VERSION OF THIS PDF DOCUMENT

This article is about using the STM32CubeIDE to develop applications for STM32 MCU's. Why develop an application using the STM32Cube IDE, and what are the advantages and disadvantages of using this method?

The STM32CubeIDE is a complete development system to develop code for almost all STM32-based MCU's from ST Microelectronics. As the name suggests, it is an Integrated Development Environment (IDE) that essentially includes the STMCubeMx GUI HW configuration tool, and a full compiler.

It can be used as a development platform for all STM32 MCU's, whether it is on a development board such as from one of ST's Nucleo or Discovery family, or a custom-designed board.

Another advantage is that it allows for much better control of the MCU. Compared to Arduino, the user is no longer limited to just what functionalities Arduino provides, and what MCU's it has been ported to.

However, this flexibility comes at a price.

For one thing, using the STM32CubeIDE locks the user into ST's ecosystem. In other words, it is geared toward the STM32 family of MCU's.

ST does have a very large selection of STM32 MCU's though, and this approach means that once an application is written, it is relatively easy to port that to another, more powerful, member of the STM32 family, should the need arise.

Another is the development code has to be in C/C++. In other words, some of the higher level HW abstractions from Arduino, for instance, are not available.

It is not as simple as calling Serial.begin or DigitalWrite functions as in Arduino. The actual UART, or GPIO port, has to be properly initialized first.

This is where the GUI configuration tool helps. Still, this also means that the user has to be quite aware of the MCU internal architecture and the actual HW design that this MCU is part of.

As will be seen later, knowledge of the appropriate ST's HAL API also helps. This Hardware Abstraction Layer Application Programming Interface can greatly ease the transition from, again, Arduino to using STM32CubeIDE. For the purpose of this article, an STM32 BluePill board will be used as the base HW. This is readily available, and serves the purpose of an introduction to STM32CubeIDE.

Other than the actual BluePill, a device programmer will also be required. For this, one of the STLInk V2 clones will be used.

Figure 1 shows an example of both the BluePill and the ST Link V2 clone. Since the intent of this article is to introduce the STM32CubeIDE, and not code development, the application example is the simple "Blinky".



Figure 1 – Example of a BLuePill and ST Link V2 device programmer clone

Configuring the Hardware

The first thing to do is to install STM32CubeIDE. This can be downloaded <u>here.</u> Note that registration is required.

After installation, you can startup the application. Select from File-New-STM32 as shown in Figure 2. After a little while, the device selection screen shown in Figure 3 will appear.

Here is where knowledge of the actual HW is required. Figure 4 shows the schematic diagram of the STM32duino BluePill module. As can be seen, the MCU is a STM32F103C8T6. So, this is what is selected.



Figure 2 – initial screen of STM32Cube IDE showing selection drop-downs for a new project

I/MPU Filters										
★ 🔂 🔂 ७		Features		Block Diagram	Docs &	& Resources	🔄 Datashee	it		🖞 Buy
art Number Search	*				NIGHT CTN (201 4+ 1	MCU				
Q					Ultra low power w	/ more performan	ce			
ore >						•				
ries >					wwww	• 120 MHz				
10 >					STI	Down to 512 KB Flas 320 KB RAM	h			
ackage >					STR. Farm	Down to 48 pins				
her 🗸					•					
rice From 0.0 to 9.54						5	7			
0 9.54										
Erom 11 to 176										
D From 11 to 176	MOU-AN	Nie Lieb 4044 James	de Diselas similas i							A. r.,
D From 11 to 176	MCUs/MF	PUs List: 1641 items	Display similar it	ems		Band Broken	Park			🔥 Ex
D From 11 to 176 1 176 ieprom From 0 to 16384 (Bytes) 16384	MCUs/MF	PUs List: 1641 items	- Display similar it Reference	ems Marketing Stat	Und Price for 10kU (US\$) 0 597	Beard Package	Flash 32 kRutes	RAM 4 kRutes	10	Ex Freq 48 MHz
D From 11 to 176 1 176 1777 1777	MCUs/MF	PUs List: 1641 items Part No * STM32F030C6 STM32F030C8	Display similar it Reference STM32F030C6Tx STM32F030C8Tx	Active	us Und Price for 10kU (US\$) 0.597 0.722	Board Package LQFP48	Flash 32 kBytes 64 kBytes	RAM 4 kBytes 8 kBytes	10 39 39	48 MHz
D From 11 to 176 1 176 ieprom From 0 to 16384 (Bytes) 0 16384 lash From 0 to 2048 (kBytes)	MCUs/MP	PUs List: 1641 items Part No STM32F030C6 STM32F030C8 STM32F030CC	Display similar it Reference STM32F030C6Tx STM32F030C8Tx STM32F030C8Tx	Active Active Active	Unit Price for 10kU (US\$) 0.597 0.722 1.1	Board Package LQFP48 LQFP48 LQFP48	Flash 32 kBytes 64 kBytes 255 kBytes	RAM 4 kBytes 8 kBytes 32 kBytes	0 39 39	Ex Freq 48 MHz 48 MHz 48 MHz 48 MHz
D From 11 to 176 1 176 aprom From 0 to 16384 (Bytes) 0 16384	MCUs/MF	PUs List: 1641 items Put No 7 STM32F030C6 STM32F030C8 STM32F030CC STM32F030CC	Display similar it Reference STM32F030C6Tx STM32F030C6Tx STM32F030CCTx STM32F030CCTx STM32F030CCTx	ems Active Active Active Active	Use Price for 104U (USS) 0.597 0.722 1.1 0.424	Board Package LQFP48 LQFP48 LQFP48 LQFP48 TSSOP20	Pash 32 kBytes 64 kBytes 256 kBytes 16 kBytes	RAM 4 kBytes 8 kBytes 32 kBytes 4 kBytes	10 39 39 37	Ex Freq 48 MHz 48 MHz 48 MHz 48 MHz
D From 11 to 176 1 From 0 to 16384 (Bytes) 0 16384 (Bytes) 1 1584 1 168 1 176 1	MCUs/MF	PUs List: 1641 items STM32F030C6 STM32F030C8 STM32F030C4 STM32F030F4 STM32F030F4	Display similar it Enformer STM32F030C6Tx STM32F030C6Tx STM32F030C4Tx STM32F030C4Tx STM32F030C4Tx	Active Active Active Active Active Active	Unit Price for 1963 (USS) 0.597 0.722 1.1 0.424 0.619	Board Package LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48	Fash 32 kBytes 64 kBytes 256 kBytes 16 kBytes 32 kBytes	RAM 4 kBytes 8 kBytes 32 kBytes 4 kBytes 4 kBytes	39 39 37 15 25	€ Ex 18 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz
D From 11 to 176 1 176 1890 From 0 to 16384 (Bytes) 0 16384 1894 From 0 to 2948 (kBytes) 0 2048 2048 2048 Am From 2 to 1184 (kBytes)	MCUs/MF	PUs List: 1641 items Pet No. C STM32F030C6 STM32F030C6 STM32F030CC STM32F030C4 STM32F030K6 STM32F030R6	Display similar it reterence STM32F030C6Tx STM32F030C6Tx STM32F030C6Tx STM32F030F4Px STM32F030F4Tx STM32F030F4Tx STM32F030F4Tx STM32F030F4Tx	Active Active Active Active Active Active Active	016 Price for 1040 (US5) 0.597 0.722 1.1 0.424 0.518 0.754	Bland Package LQFP48 LQFP48 LQFP48 LQFP48 TSSOP20 LQFP32 LQFP32 LQFP34	Fesh 32 kBytes 64 kBytes 256 kBytes 16 kBytes 32 kBytes 64 kBytes	RAM 4 kBytes 8 kBytes 32 kBytes 4 kBytes 4 kBytes 8 kBytes	39 39 37 15 25 65	Ex Free 48 MHz
From 11 to 176 rpmm 176 genom 165384 (Bytes) 0 16384 (Bytes) 2048 2048 am From 2 to 1184 (kBytes) 2 1184	MCUs/MF	Pus List: 1641 items STM32F030C6 STM32F030C8 STM32F030C8 STM32F030F4 STM32F030F4 STM32F030R6 STM32F030R8 STM32F030R8	Display similar it Contraction Co	Active Active Active Active Active Active Active Active	us Beef Proce for 1840 (US5) 0.597 0.722 1.1 0.424 0.518 0.754 1.21	Grand CoPP48 LOPP48 LOPP48 LOPP48 TSS0P29 LOPP32 NSCLE_32F85_LOPP64	Fash 32 kBytes 64 kBytes 256 kBytes 16 kBytes 32 kBytes 64 kBytes 24 kBytes	RAM 4 KBytes 32 kBytes 4 KBytes 4 kBytes 8 kBytes 23 kBytes	39 39 37 15 25 55 51	€ Ex 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz
Frem 11 to 176 tapsom From 0 to 15384 (Bytes) 15384 tash From 0 to 2548 (Ebytes) 2048 am From 2 to 1184 (Ebytes) 24 18 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 5 4 5 4 5 4 5 4 5 4 5 4 5 5 5 6 6 6 7 7 7 6 7 7 7 7	MCUs/MF	PUs List: 1641 items STM32F030C6 STM32F030C8 STM32F030C8 STM32F030C8 STM32F030C8 STM32F030K6 STM32F030K6 STM32F030K8 STM32F030K8 STM32F030K8	Display similar it Deterrine STM32F030CETx STM32F030CETx STM32F030CETx STM32F030FETx STM32F030FETx STM32F030FETx STM32F030FETx STM32F030FETx STM32F030FETx STM32F030FETx	Active Active Active Active Active Active Active Active Active	a Dest Price for 1942 (US5) 0.597 0.722 1.1 0.424 0.518 0.754 1.21 0.97	Beard Pastage LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP48 LQFP49 LQFP44 LQFP40 LQFP46 LQFP48 LQFP48	Fash 32 kBytes 64 kBytes 256 kBytes 16 kBytes 32 kBytes 64 kBytes 256 kBytes 256 kBytes	RAM 4 kBytes 8 kBytes 32 kBytes 4 kBytes 4 kBytes 8 kBytes 32 kBytes	39 39 37 15 25 55 55 55	€ Ex 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz 48 MHz
Frem 11 to 176 176 sprom From 0 to 15384 (Bytes) 16384 Table From 0 to 2948 (kBytes) 2048 Tan From 2 to 1184 (kBytes) 1184 Tan From 2 to 1184 (kBytes) 1184 Trees From 2 to 100 (MHz)	MCUs/MF	PUs List: 1641 items	Display similar it Reference STMO2F0300GTx STM03F0300GTx STM03F0300GTx STM03F0300F4Px STM03F030F4Px STM03F03F04Px STM03F03F0F4Px STM03F03F7 STM03F0	Active Active Active Active Active Active Active Active Active	as 6ad Proce for 1930 (055) 0.597 0.722 1.1 0.424 0.518 0.754 1.21 0.97 4.412	Brand Providing LOFP48 LOFP48 LOFP48 LOFP48 LOFP48 LOFP48 LOFP48 LOFP64 LOFP48 LOFP64 LOFP48 LOFP64	Flash 32 kBytes 64 kBytes 256 kBytes 16 kBytes 32 kBytes 256 kBytes 256 kBytes 16 kBytes 256 kBytes 16 kBytes	PAM 4 kBytes 8 kBytes 32 kBytes 4 kBytes 8 kBytes 32 kBytes 32 kBytes 4 kBytes 4 kBytes	39 39 37 15 25 55 51 39	€ Ex 48 MHz 48 MHz
From 11 to 176 (aprom From 0 to 16384 (Dytes) 16384 (Lbytes) 16384 (Lbytes) 2048 am From 2 to 1184 (L6bytes) 2 1184 4 6 184 6 2 184 6 6 184 800	MCUs/MF 2 2 2 2 2 2 2 2 2 2 2 2 2	PUs List: 1641 Rems STM32F030C6 STM32F030C6 STM32F030C6 STM32F030C4 STM32F030F4 STM32F030F4 STM32F030F8 STM32F030F8 STM32F030F8 STM32F031C4 STM32F031C6 STM32F031C6	Display similar in STM32F 000GTx STM32F 000GTx STM32F 000GTx STM32F 000GTx STM32F 000GTx STM32F 000GTx STM32F 000GTx STM32F 000GTx STM32F 000GTx STM32F 000GTx	Active Active Active Active Active Active Active Active Active Active	as 0x4 Pres for 184/ 0153 0.597 0.722 1.1 0.424 0.518 0.754 1.21 1.21 0.97 1.013	Need Person LOFP48 LOFP48 LOFP48 TSSOP20 LOFP4 LOFP64 LOFP64 LOFP64 LOFP64	Tenh 32 kBytes 64 kBytes 256 kBytes 32 kBytes 64 kBytes 64 kBytes 65 kBytes 64 kBytes 256 kBytes 32 kBytes	RAM 4 kBytes 3 kBytes 3 kBytes 4 kBytes 8 kBytes 3 kBytes 4 kBytes 4 kBytes 4 kBytes	39 39 37 15 25 55 51 39 39	€ Ex 48 MHz 48 MHz
Frem 11 to 176 176 sprom From 0 to 15384 (Bytes) 16,344 Tash From 0 to 2548 (kBytes) 2048 Tash From 2 to 1184 (kBytes) 2048 Tash From 2 to 1184 (kBytes) 1184 Tesper From 2 to 1184 (kBytes) Tesper 1184 Tesper 1184 Tesper 1184 Tesper 1184		PUs List: 1641 items STM32F030C6 STM32F030C6 STM32F030CC STM32F030CC STM32F030F4 STM32F030F4 STM32F030F8 STM32F030F8 STM32F031C6 STM32F031C6 STM32F031C6 STM32F031C6	Complex similar it Complex similar it STM2F930C5Tx STM2F930C5Tx STM2F930C5Tx STM2F930C5Tx STM2F930C5Tx STM2F930C5Tx STM2F930C5Tx STM2F931C4Tx STM2F931C4Tx STM2F931C4Tx	Active Active Active Active Active Active Active Active Active Active Active	test free for 160/ (015) 0.597 0.722 1.1 0.424 0.518 0.754 1.21 0.97 1.013 0.776	Read Product LOPP48 LOPP48	7845 32 kBytes 64 kBytes 16 kBytes 32 kBytes 54 kBytes 256 kBytes 16 kBytes 32 kBytes 32 kBytes 32 kBytes	RAM 4 KBytes 8 KBytes 4 KBytes 4 KBytes 8 KBytes 22 KBytes 4 KBytes 4 KBytes 4 KBytes 4 KBytes	39 39 37 15 25 55 51 39 39 39 20	€ Ex 48 MHz 48 MHz
From 11 to 176 176 icprom From 0 to 15384 (Bytes) 15384 bash From 0 to 2648 (kBytes) 2048 am From 2 to 1184 (kBytes) 1184 req. From 24 to 800 (MHz) 4 req. spon 400 represel V		PUs List: 1641 Rems STM32F030C6 STM32F030C6 STM32F030C6 STM32F030C6 STM32F030C6 STM32F030K6 STM32F030K6 STM32F030K6 STM32F030K6 STM32F031C6 STM32F031C6 STM32F031C6 STM32F031C6	Display similar it STM32F 03006Tx STM32F 03006Tx STM32F 03006Tx STM32F 03004Fx STM32F 03004Fx STM32F 03004Tx STM32F 03104Tx STM32F 03104Tx STM32F 03104Tx STM32F 03104Tx STM32F 03104Tx	Active Active Active Active Active Active Active Active Active Active Active Active Active Active	Bit Over for 1800 (USS) 0.557 0.722 1.1 0.424 0.518 0.754 1.21 0.97 1.013 0.776 0.711	Deed Potage LQFP48 LQFP48 LQFP48	Fash 32 kBytes 64 kBytes 64 kBytes 256 kBytes 16 kBytes 32 kBytes 64 kBytes 16 kBytes 16 kBytes 32 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes	total 4 KBytes 8 KBytes 32 KBytes 4 KBytes 4 KBytes 4 KBytes 4 KBytes 4 KBytes 4 KBytes 4 KBytes	0 39 37 15 25 55 51 39 39 20 15	€ Ex Free 48 MHz 48 MHz
Prem 11 to 176 Trom 11 to 176 Trom 7 to 1558 (Bytes) To 344		PUs List: 1641 items STN422F030C6 STN422F030C8 STN422F030C8 STN422F030C8 STN422F030C8 STN422F030C8 STN422F030C8 STN422F031C6 STN422F030C7 STN422F031C6 STN422F031C6 STN422F031C6 STN422F031C6 STN422F031C7 STN42F031C7 STN42F031C7 STN42F031C7 STN42F030C7 S	Copplay similar it STM2FP 900CFX STM2FP 900CFX STM2FP 900CFX STM2FP 900CFX STM2FP 900FX STM2FP 900FX STM2FP 900FX STM2FP 910FX STM2FP 910F4PX STM2FP 91F4PX STM2FP 91F4PX STM2FP 91F4PX STM2FP 91F4PX	twentry file Active	as Dear Price for 1962 (015) 0.597 0.722 1.1 0.424 0.518 0.754 1.21 1.21 0.71 0.716 0.711 0.715	Read CPR48 LQFP48 LQFP48	Facto 32 kBytes 64 kBytes 255 kBytes 16 kBytes 26 kBytes 26 kBytes 26 kBytes 32 kBytes 32 kBytes 32 kBytes 32 kBytes 32 kBytes	4 kBytes 8 kBytes 32 kBytes 4 kBytes 8 kBytes 32 kBytes 32 kBytes 32 kBytes 4 kBytes 4 kBytes 4 kBytes 4 kBytes	0 39 37 15 25 55 51 39 39 20 15 15	€ Exp 48 MHz 48 MHz
O From 11 to 176 From 11 to 176 From 7 tron 0 to 16384 (Bytes) From 0 to 2048 (Etytes) Tash From 0 to 2048 (Etytes) From 2 to 1184 (Etytes) From 2 to 1184 (Etytes) From 2 to 800 (MHz) From		*Us List: 1641 fierns STM26790056 STM26790056 STM26790056 STM26790057 STM26790057 STM26790056 STM26790166 STM26790166 STM26790166 STM26790166 STM26790164	Complay similar in to an excess STMASP 03000Th STMASP 03000Th STMASP 03000Th STMASP 03000Th STMASP 03000Th STMASP 0300Th STMASP 03105Th STMASP 03105Th STMA	Active Ac	as Cost Pres for 1847 (USS) 0.597 0.722 1.1 0.424 0.518 0.754 1.21 0.97 1.013 0.776 0.711 0.755 0.733	Deed Potage LGFP48 LGFP48 LGFP4708 LGFP48	7 and 32 kBytes 64 kBytes 256 kBytes 16 kBytes 32 kBytes 32 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes 16 kBytes	4 kBytes 8 kBytes 32 kBytes 32 kBytes 4 kBytes 4 kBytes 4 kBytes 4 kBytes 4 kBytes 4 kBytes 4 kBytes 4 kBytes	0 39 37 15 25 55 51 39 39 20 15 15 23	€ Exp res 48 MHz 48 MHz
D From 11 to 176 from 500 to 15584 (B)(tes) from 20 to 15584 (B)(tes) from 20 to 1548 (B)(tes) from 20 to 1184 (B)(tes		*Us List: 1611 8ems STMASF000C6 STMASF000C8 STMASF000C8 STMASF0008 STMASF0008 STMASF0008 STMASF0008 STMASF0008 STMASF0104 STMASF0116 STMASF0116 STMASF0116 STMASF0116	Constant and a second s	Active Ac	Dest Price for 1960 (015) 0.597 0.722 1.1 0.424 0.518 0.754 1.21 0.97 1.013 0.776 0.711 0.755 0.733 0.776	Read Prison LQFP48 LQFP48 UCP49 TISSOP20 UTOFPH08 IFORDARIA	7 and 20 kBytes 64 kBytes 256 kBytes 16 kBytes 256 kBytes 26 kBytes 26 kBytes 22 kBytes 22 kBytes 22 kBytes 22 kBytes 22 kBytes 23 kBytes 24 kBytes 24 kBytes 25 kBytes 26 kBytes 26 kBytes 26 kBytes 27 kBytes 27 kBytes 27 kBytes 26 kBytes 27 kByte	4 KBytes 8 KBytes 22 KBytes 4 KBytes 4 KBytes 32 KBytes 4 KBytes	39 39 37 15 25 55 51 39 39 39 20 15 15 15 23 23	€ En 48 MHz 48 MHz
Prent 11 to 176 From 11 to 176 Septem From 0 to 15384 (Bytes) 0 1184		Va List: 1641 terms STM26790056 STM26790056 STM26790056 STM26790056 STM26790056 STM26790066 STM26790066 STM26791066 STM26791165 STM26791165 STM26791164 STM26791164	Complay similar it to an excess STMASP 03000Th STMASP 03000Th STMASP 03000Th STMASP 03000Th STMASP 03000Th STMASP 03000Th STMASP 0300Th STMASP 0300Th STMASP 03154h STMASP 03164h STMASP 0316	Active Active	as Continues for 1800 (USS) 0.597 0.722 1.1 0.424 0.518 0.754 1.21 0.97 1.013 0.776 0.711 0.755 0.733 0.776	Deed Potage LQFP48 LQFP48	7 wat 22 & Mbytes 266 Mbytes 266 Mbytes 276 Mbytes 2	4 kBytes 8 kBytes 32 kBytes 32 kBytes 4 kBytes	39 39 37 15 25 55 51 39 39 20 15 15 23 23	▲ Ex 48 Metz 48 Me

Figure 3 – Device selection screen



Figure 4 – STM32Duino BluePill schematic

The next screen requests a name for the project, and also some information about the type of project. In this case, as shown in Figure 5, the project will be an executable called Blink, and it will be done using C.

After a while, and answering "yes" to the pop-up about STMCube, the screen shown in Figure 6 will appear. This is the device pinout view screen. It allows the user to choose the function of the MCU pins.

The pins of the STM32 MCU, like most other MCU's, can be configured for many uses such as GPIO, ADC, Timer and others. Here again, knowledge of the actual HW is needed to properly configure the pins for their intended use in the HW.

First, select the proper clock source for the MCU. By default, the MCU uses its internal RC oscillator. However, in the STM32duino BluePill, the clock is actually an external crystal.

Again referring to figure 4, the BluePill external crystal frequency is 8 MHz. In keeping with the most commonly used clock frequency for the BluePill, the internal clock frequency will be set to 72 MHz.

First go to the Pinout & Configuration tab, and select RCC; this stands for Reset and Clock Control. Now in the panels that appear, choose the High Speed Clock (HSE) to be "Crystal Ceramic Resonator".

Next, select which pins the Crystal is going to be attached to. Even though the schematics in figure 4 label these pins as OSCIN and OSCOUT, they are, in fact, PD0_OSC_IN and PD1_OSC_OUT. Just left-click on PD0 and PD1, and select their functions to be OSC_IN and OSC_OUT respectively. This completes the setting of the pins for external crystal usage.

The next part is to set up the clock frequency. Click on the Clock Configuration tab, and a screen like that shown in figure 8 will appear. Values are entered directly in the various boxes.

Since the external crystal clock is 8MHz, just make sure that this is set to 8. Next select HSE, or High-Speed External, in the PLL source MUX. Then, set the PLL MUL to 9.

This essentially multiplies the input clock by 9, giving it a value of 72MHz. Now, set the System Clock Mux to PLLCLK. That is, tell the MCU that its clock source will be the output of the PLL.

There are many other options. For example, the CSS can be enabled. CSS stands for Clock Security System. If enabled, then a non-maskable Interrupt is generated if the external clock fails. Otherwise, the MCU switches to use its HSI, or High Speed Internal, clock. In this example, this will not be enabled. The confusing part is the box will show "Enable CSS", but it is actually not enabled.

If enabled, the box will show "CSS Enabled". Finally, the APB1, or Advanced Peripheral Clock 1, prescaler should be set to /2 since the maximum APB1 clock frequency is 36MHz. This is the clock source for the UART, for example. The other options allow the peripherals to be clocked at frequencies other than core clock.

This can be used to reduce power consumption, for instance. However, in this case, the rest will be left as is. Figure 8 actually shows the proper values as just described.

Since the application requires blinking the built-in LED, the pin driving the internal LED needs to be set up now. Looking at the schematic diagram of figure 4 once again, it is seen that the internal LED is driven by PC13. So, this pin has to be set to be a GPIO output pin.

Go back and click on the Pinout & Configuration tab. On the pinout diagram, left click on the PC13 pin, and select GPIO_Output. Right click on the same pin, and select "Enter User Label". Enter a name such as "LED" for this pin.

Now, select System View on the main window, and select GPIO. The screen will look like that shown in Figure 9. In the GPIO Mode and Configuration window, various settings for the GPIO pin can be selected.

As shown in Figure 9, the initial output is Low, the mode is push-pull and there is no pull-up or pull-down. That should be fine for this application, but just be aware that these settings can be changed to suit the application. This completes the required settings for this application.

workspace_1.3.0 - STM32CubeIDE	STM32 P	roject	_		1											5 X
File Edit Source Refactor Navigate Search	Setup STM32	! project		IDE	\$+\$+ d	0									Quick Access	8 🖬 🖬
A Project Explorer II ► 2 > III Text	Project Project Nar ⊇ Use defai Location Coptions Targeted ⊕ STM3	ve Bind CrUser/Shann/STMJ CrUser/Shann/STMJ Cr+ Binary Type Able O Static Library Project Type Crube O Empty Kack Ne	XubeDE/workspace, 13.0	Brows												
		Problems SS @ T	asks 💿 Console 📰 Propert	ies			1	▽ □ □	Build Analyzer	🕄 🛓 Static Sta	ick Analyzer					
		Description		Resource	Path	Location	Ђре		Memory Regions	Memory Detail	s Fadaddau	Circ.	f	Bed	Harry (N)	
									Negion	Julit autoress	and address	3126	1166	Used	Usege (/a)	
														1		_
₽ Type here to search		O Ħ	= 😒 🤹 🧕	<u></u> 74 (§) 💽 🐖	IDE									• ♥ // ¢0) 6:46 4/26	AM /2020 22

Figure 5 – Project information screen



Figure 6 – Pinout view of the STM32F103C8Tx MCU



Figure 7 – Pinout & Configuration tab for selecting external clock source





🔁 • 🗟 🔞 🕲 • 🗞 • 🛍 🔪 d	i @ • ◎ • € • @ • ‡	• O • G • 🤷 • 🖄	(i) # [오] * 5]	• \$\$ \$\$ • \$	0 121								Quick Acc	ess E	8 🗟 🔤
Project Explorer C Project Explorer C	Pinout /	լդ main.h լդ stm <i>s2</i> rlox_haij 3. Configuration	iont.h	Clock	Configurati tional Sof	ion tware	✓ Pinout	Project	Manager				Tools		
> pi Includes <pre> <pre> <pre> <pre> </pre> <pre> <pre> <pre> <pre> </pre> </pre> <pre> <pre> <pre> <pre> <pre> </pre> </pre> </pre> <pre> <pre> <pre> <pre> <pre> <pre> <pre> </pre> </pre> </pre> </pre> <pre> <pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre>	Q (() Categories A->Z System Core ~	Group By Peripherals	GPIO Mo	de and Configuration	n		J		-	😂 Pinout viev	w Syster	n view			2
) () () sha3tor,tA) () () ()) () () () () () () () () () () () () ()		GPIO RCC Search Signals Search (crif+F) Pin Name Signal on Pin Cr PC13-TAM n/a Lo	PIO outp GPIO m v Output F	ode GPIO Pull us No pull-up	Maximum . High	Show only Modified	Pins								
	Timers > Connectivity > Computing > Middleware >	PC13-TAMPER-RTC Configur GPIO output level GPIO mode GPIO Pull-up/Pull-down Maximum output speed User Label	ition :	Low Output Push Pu No pull-up and n High	io pull-down			0			132F103C8TX LOFP48		0	~	
	Problems 23 2 Tasks O Items Description	Console Console Properties Resource	Path	Location	Туре	= - 4 	Build Analyzer Blink.elf - /Blin Memory Region Region BRAM BRAM	r 22 al Static Stat nk/Debug - May 75 Memory Details Start address 0x2000000 0x0800000	ck Analyzer 1, 2020 5:18:48 / End address 0x20005000 0x/0010000	AM Size 20 KB 64 KB	Free 18.45 KB 59.55 KB	Used 1.55 KB 4.45 KB	Usage (%) 7.73% 6.95%		•••

Figure 9 – System view screen

Code

So, what does all the configuration in the previous section do? It allows the automatic generation of initialization code for setting up the clock and GPIO.

Just go the "Project" tab, and click on "Generate Code". A pop-up progress window will appear, and after it is done, navigate to the "Project Explorer" tab.

From there go to "core", then to "Src", and click on main.c. This will open up the main.c file in the main window. As can be seen in figure 10, lots of code has been generated already.

Scrolling down this window to the *int main(void)* function, it is seen that it already calls three functions: *HAL_Init()*, *SystemClock_Config()*, and *MX_GPIO_Init()*. The actual functions are defined further down in main.c. These are the HW setup functions that were automatically generated based on the user configurations entered previously.

Also note that there are various sections with comments that start with something like /* USER CODE BEGIN WHILE */, and /* USER CODE END WHILE */. The space between any comments that are bracketed by comments that start with "USER CODE ..." is where the user is expected to enter the application-specific code. This is shown in figure 11.

In this article, some code will be entered in the sections defined in the previous paragraph. Before getting to that, open the main.h file, and locate this section:

```
/* Private defines ------*/
#define LED_Pin GPIO_PIN_13
#define LED_GPIO_Port GPIOC
/* USER CODE BEGIN Private defines */
```

```
/* USER CODE END Private defines */
```

Using the "LED" label provided earlier, the code generator has already defined some names to GPIO_PIN_13 and the port it is on. In this case, LED_Pin is GPIO_PIN_13, and LED_GPIO_Port is GPIOC.

Go back to main.c, and insert the following two lines between /* USER CODE BEGIN WHILE */, and /* USER CODE END WHILE */, as shown in figure 12.

```
HAL_Delay(500);
HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
```

HAL GPIO TogalePin

The first line is a delay of 500ms, and the second line simply toggles the LED pin as defined earlier. Since this is in the main loop, it is repeated forever, and thus the LED blinks forever.

The more important part of the code is the actual functions that were called in the main loop. These functions are part of the HAL API that is already included in the application that was just developed.

Understanding this HAL API, and knowing what is available, and how to call the different HAL functions goes a long way toward making code development for STM32 MCU's easier and faster. <u>Here</u> is a link to the HAL API.

As a side note, notice how this blink function is different from the typical Arduino blink implementation due to the single call to the *HAL_GPIO_TogglePin()* function.

Below is the actual section on this function as lifted from the HAL user manual.

Function name	void HAL_GPIO_TogglePin (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)
Function description	Toggles the specified GPIO pins.
Parameters	 GPIOx: Where x can be (AK) to select the GPIO peripheral for STM32F429X device or x can be (AI) to select the GPIO peripheral for STM32F40XX and STM32F427X devices. GPIO_Pin: Specifies the pins to be toggled.
Return values	□ None:

Scanning through this section of the HAL User Manual, there are many GPIOrelated functions, including this one, that are not available in the native implementation of Arduino.

[i] 🕲 • %	• 📾 🔪 🤞	10 - 53 -	C • G • 1	-0.9	- 10 11 -	1 3	🗐 n 191 -	0.00.	0.101	D							Quick A	Access	EP
t Explorer SS	- 0	Blink.ioc	a main.c 3	8 main.h	stm32f1xx	_hai_conf	Uh .												E
nk Binarias Includes Core Includes Incl	E & v	1 // * USI 2 // * USI 3 * * * * * * * * * * * * * * * * * * *	R CODE BEGI ile second il contents il classifier il classifie	N Header */ : main.c : Pain pr copy; Copy: Hain pr copy; Copy: Main pr eserved.c/cen may obtain a pr Header */ N Includes * N Includes * N Includes * N PD */ PTD */ P	pgram body ppt (c) 2020 ter>c/h2> n Licensed by is use this fit copy of the mesource.org/	STNicrov	electronics. er 850 3-Clau tin complete sti /\$55-7-Clau	se license, nce with the e											
		Problems	Tests	🖸 Console 🛅	Properties					*	Build Analyze	er 23 al Static Sta	ick Analyzer					et 1	φ 6
		0 items									Blink.elf - /Bl	ink/Debug - May	1, 2020 5:18:48	AM					
		Description			Resou	rce	Path	Location	Type		Memory Regio	ins Memory Detail	5						
											Region	Start address	End address	Size	Free	Used	Usage (%)		
											E RAM	0x20000000	0x20005000	20 KB 64 KB	18.45 KB	1.55 KB	7.73%		
														0.700700					

Figure 10 – main.c

Image: Section of the section of th	8 H
Repertspoor :: Image: intermediate in	e 111 (*)
Bink 619 ^{-/*} Bink 2 %binf 619 ^{-/*} > 2 %binf 619 ^{-/*} %binf 619 ^{-/*} > 3 %binf 619 ^{-/*} %binf 619 ^{-/*} > 4 %binf %binf %binf %binf > 3 %binf %binf %binf %binf > 3 %binf %binf <t< td=""><td>84 ®</td></t<>	84 ®
<pre>> B unback.mccumn 70 > B unback.mccumn 71 > B</pre>	
<pre>> bread > bread ></pre>	
😰 Peoblems 🕄 🕘 Tasks 😳 Consulte 📄 Properties 🏦 🦆 🗢 🗇 📴 Beild Analyzer 🕄 🛓 Static Stack Analyzer	
D Rems Path Lection Type Bink.eff.~91m// DebugMay 1, 202 51:84 J Marce Feature Lection Type Bink.eff.~91m// DebugMay 1, 202 51:84 J Marce Lection Type Bink.eff.~91m// DebugMay 1, 202 51:84 J Marce Lection Type Bink.eff.~91m// DebugMay 1, 202 51:84 J Marce Lection Type Marce Debug Debug Debug <td></td>	
₩F/LASH 0x0000000 0x00010000 64 K3 55.5 K3 4.45 K3 55.5 K3 55	
Wittable Smart Insert 291:81:1124 Image: P Type here to search O Image: Image: P Image: Image: P	3

Figure 11 – main.c areas where the user can enter code

workspace_1.3.0 - Blink/Core/Src/main.	hain c - STM32CubeIDE		- 0	×
	@ @ + @ + @ + Q + Q + Q + @ @ # * J # @ @ # ! D + @ + D + D + D + D + D + D + D + D +	Quick	Access	E2) ===
Project Explorer 🐹 📟 🗖	I Blinkice 🖟 main.e 😢 🖗 main.h 🛞 stm32floc,hal_conf.h			
Constant Sector Constant Sector Constant Sector Constant Sector Sec	<pre>v * '' i to st definition '/ USE (COE BEGN 1 */ '/ USE (COE BEGN 1 */ '/ USE (COE BEGN 1 */ '/ '/ USE (COE BEGN 5yslet */ '/ '/ USE (COE BEGN 5yslet */ '/ '/ USE (COE BEGN 5yslet */ '/ '/ USE (COE BEGN 1 */ '/ '/ USE</pre>			
	C		>	
	(2) Posteres 3 Taks Q Consele 22 D Popertes 4 9 9 3 II II = 1 k C □ + 1 + 1 + 2 + 2 + 2 + 2 + 2 + 2 + 2 + 2		1 T T	
	17/53/11 **** Incremental Build of configuration Debug for project Blink **** Blink.df /			
	arronze-sebi-siz Blink-elf Region Stataders Endaders Size Free	Used Usage (%)		
	4676 28 1572 6265 1877 81ink.elf Pinished building: effektivis.size.stdout # 5145 KB	1.55 KB 7.73%		
	17:33:34 Build Finished. 0 errors, 0 warnings. (took 2s.78Ins)			
	×			
	Writable Smart Insert 105:	1:2805		
+ P Type here to search	o # 📕 💁 🚺 💶 🜌 🦉 🤌 😁	^ Ÿ	(化 વ0) 6:00 PM 5/1/2020	23

Figure 12 – Added code to while (1) main loop

Compiling and loading

To compile the code, simply go to the Project tab, and select Build All. Assuming there are no compile errors move to the Debug folder and locate the .bin file.

As shown in figure 13, this is Blink.bin. This is the file that has to be loaded to the actual STM32 MCU. The actual loader can be the STM32 ST-Link Utility.



Figure 13 – Locating the .bin file

The actual process of loading a binary file into the STM3Bluepill is fully described in many other places, so I'll just provide a summary.

Here are the steps:

- 1. Connect the following pins of ST-Link V2 to the STM32duino: 3.3V, SWDIO, SWCLK and GND.
- 2. Set the STM32duino to DFU, Device Firmware Upgrade, mode by moving the BOOT_0 jumper to the 1 position.

- 3. Plug in the ST-Link V2 and start the utility. A screen looking somewhat like figure 14 will appear.
- 4. Connect to the STM32duino, and then flash the code. Address of 0x800000 is fine as is.

To test, put the STM32duino back to user mode, and simply reset by depressing the RESET pushbutton. The LED on the board should flash, indicating a successful code download.

stm32 st-link Utility			_		Х
File Edit View Target ST-LINK External Lo	ader Help				
🖴 🖥 🖕 🤹 🏈 🛸 📄					
Memory displat Address: 0x08000000 V Size 0x1258	Data Width: 32 bits 🗸	Device Device ID Revision ID Flash size			odata
Device Memory				Liveo	poate
20:46:47 : Device ID:0x410 20:46:47 : Device flash Size : 128KBytes 20:46:47 : Device family :STM32F 10xx Medium-density 20:47:14 : Memory programmed in 0s and 453ms. 20:47:14 : VerificationOK 20:47:14 : Programmed memory Checksum: 0x0006957 20:47:38 : Disconnected from device. 20:47:38 : Disconnected from device. 20:47:38 : Connection to device is lost: check power su 20:47:38 : If the target is in low power mode, please en	7C Ipply and debug connection. nable "Debug in Low Power mode" option fr	om Target->settings menu.			<
Disconnected	Device ID :	Core State : No Memory L	paded		_

Figure 14 – ST-Link utility screen

Conclusion

It has been a rather long and tedious process to get the STM32duino to simply blink its LED. Obviously, for this particular task,

Although its not necessarily the best way to achieve a blinking LED, I hope this example has illustrated how to set up and use the STM32CubeIDE

Once you understand this process, you can use any STM32 MCU for any HW design, with full access to its internal peripherals. This really opens up new possibilities that are not typically offered by more restricted development systems.

Written by Shawn Litingtun who is one of the <u>experts</u> available to help you with your product inside the <u>Hardware Academy</u>.

