

# **Development of code to write to the microcontroller**

Currently microcontrollers are programmed in high-level languages, such as C/C++ or Java. However, in embedded *firmware* programming, the C language is widely used. The reason for this is that C is a language that talks directly to the *hardware*. While the Assembly language is closer to the *hardware*, because its sequence of commands are like machine code that generates hexadecimal instructions, the C language has structures that are mapped directly into machine instructions.

During the firmware development process, we write a sequence of instructions into the microcontroller's memory through an Integrated Development Environment (IDE) in an embedded language, such as C, compiling and linking the code modules with libraries.

# 1. Cloning a repository

You can review the process of cloning a repository in the **Software Development Environments for Microcontrollers** content. In this tutorial we will clone the example project "**PushButton\_LoRaWAN**" available in the HT Micron GitHub repository through the branch "**git clone --single-branch --branch SDK https://github.com/htmicron/htlrbl32l.git**". As seen in the image below.



Image 1: Git Clone. Source: The author.

To view the cloned branch files, type the "LS" command.



Image 2: Viewing Files in Windows PowerShell. Source: The author.

# 2. Project Code

After the code cloning process, use the Wise Studio IDE software to run it. To perform this procedure, open the "File" tab in Wise Studio, click on "Open projects from file System". Next click "Directory", select the downloaded or cloned folder, and then click Finish.

Import Projects from File System or Archive         This valued analyzes the content of your folder or archive file to find projects and import them in the IDE.         Import source       Import source         type filer text         Folder       Import as         Instruction on envy imported projects upon completion       Import source         Import for project upon completion       Import as         Import for instel projects       Import for folder         Import for projects upon completion       Import as         Import for instel projects       Import for folder         Import for onside projects       Import as         Import for instel projects       Import instemants         Import for instel projects       Import instemants         Import for instemants       Import instemants         Import instemants       Import instemants
Import source   Import source   Import as   Folder   Import as   Ocean newly imported projects upon completion   Use installed project configurators to:     Import as   Organizar * Nova pasta   Norme   Data de modificação Tipo   Tipo   Immonto   Import as   Organizar * Nova pasta   Norme   Data de modificação Tipo Tamanho Intanéno Intended arquivos Importanto (RaWAN) Bueto of noted arquivos Importanto (RaWAN) Bueto (RaWAN)
type filter text     Folder     Import as     Calcase newly imported projects upon completion   Use installed project configurations to:     Search for nested projects   Search for nested projects natures     Vorking sets   Organizar *   Norme   Data de modificação   Tipo   Immanho   Incall Add project to working sets   Vorking sets   Organizar *   Norme   Data de modificação   Tipo   Immanho   Incall Add Project to working sets   PushButton_LoRaWAN   30/06/2022 20:39   Pasta de arquivos     PushButton_LoRaWAN   24/06/2022 15:01   Pasta de arquivos
Folder     Import as       Desdect All       0 of it exterted       1 of it exterted
Image: Construction of the stability of the st
Close newly imported projects upon completion   Use installed project configurators to:   Starch for nested projects   Detect and configure project natures   Organizar * Nova pasta   Working sets   Add project to working sets   Working sets   Data de modificação   Tipo   Tamanho   LeRaWANN_TagolO_DashBoard   24/06/2022 15:01   Pasta de arquivos
Consistent with product objects   Use installed robusts to its   Search for nested projects   Detet and configure project natures   Organizar * Nova pasta   Working sets   Add project to working sets   Working sets   Data de modificação   Tipo   Tamanho   PushButton_LoRaWAN_Bluetooth 24/06/2022 15:01 Pasta de arquivos Pasta de arquivos PushButton_LoRaWAN-Bluetooth 24/06/2022 15:01 Pasta de arquivos
Image: Search for nested projects.       Organizar <ul> <li>Nova pasta</li> <li>Detect and configure project natures</li> <li>Norme</li> <li>Data de modificação</li> <li>Tipo</li> <li>Tamanho</li> </ul> Norme       Data de modificação       Tipo       Tamanho         Mording sets       LoRaWAN_TagolO_DashBoard       24/06/2022 15:01       Pasta de arquivos         Image: PushButton_LoRaWAN       30/06/2022 20:39       Pasta de arquivos         Image: PushButton_LoRaWAN-Bluetooth       24/06/2022 15:01       Pasta de arquivos
Working sets     Organar Nova parta     Data de modificação     Tipo     Tarmanho       Working sets     Morie     Data de modificação     Tipo     Tarmanho       Working sets     LoRaWAN_TargolO_DashBoard     24/06/2022 15.01     Pasta de arquivos       PushButtor_LoRaWAN-Bluetooth     24/06/2022 15.01     Pasta de arquivos
Nome     Data de modificação     Tipo     Tamanho       Working sets     LoRaWAN_TagolO_DashBoard     24/06/2022 15:01     Pasta de arquivos       PushButtor_LoRaWAN     30/06/2022 15:01     Pasta de arquivos       PushButtor_LoRaWAN-Bluetooth     24/06/2022 15:01     Pasta de arquivos
Working sets:       Image: LoRaWAN_Tagol0_DashBoard       24/06/2022 15:01       Pasta de arquivos         Image: PushButtor_LoRaWAN       30/06/2022 15:01       Pasta de arquivos         Image: PushButtor_LoRaWAN-Bluetooth       24/06/2022 15:01       Pasta de arquivos
Image: Constraint of the second se
Image: PushButtor_LoRaWAN-Bluetooth     24/06/2022 15:01     Pasta de arquivos
Pastar PushButton_LoRaWAN

Image 3: Selection of the folder with the cloned project code. Source: The author.

The file will be available in **Wise Studio's** "project explorer" tab. Then open the main structure of the code in the "Application" folder.

🖹 workspace - WiSE Studio	
File Edit Source Refactor Navigate Search Project Run Window Help	
<u>□□ ▼ □□ □ ● ▼ ⑤ ▼ ⑥ 副 ▼ @ ▼ @ ▼ @</u> ▼ ! 株 ▼ O ▼ <b>G ▼ </b> ◎ <i>♥</i> ▼   剛 回 + 1 包 ▼ 同 ▼ ♡ や ▼ ○ ▼   団	Quick A
Deproject Explorer 🗉 📄 😇 🐨 🗖	
V 🖙 HTLRBL32L-PushButton LoRaWAN	
> @ Includes	
✓ ie-Application	
> 🗟 bluenrg_lp_it.c	
> 🔒 gpio.c	
> 强 HT_push_button.c	
> 🔒 lorawan_setup.c	
> Be main.c	
> is the	
> is spic	
) is syscalls.c	
> is uarc	
> Debug	
> an Orivers	
HTIRBI32L-PushButton LoRaWAN.cfg	
HTLRBL32L-PushButton_LoRaWAN.elf.cfc	
🖻 HTLRBL32Lld	
≥ sip_httrbl321.xml	
> 📁 HTLRBL32L-PushButton_LoRaWAN_Bluetoot	
🕼 Problems 🕸 🖉 Tasks 🖾 Properties	

Image 4: Wise Studio Project Explorer. Source: The author.

#### **2.1 How the code works.**

The main functionality of the code is to activate the LoraWan communication protocol using a push button. When the push button is pressed, an interrupt "wakes up" the device and sends a payload in data frequency. This process is very useful in the IoT environment, as most devices need to save power and battery. However, we will show the main files that deal directly with the **LoraWan** protocol interrupt through the PushButton. The source code "main.c" initializes all system functions, we can see in the image below:

- A) GPIO initialization
- **B)** Initialization of the UART, it enables the serial communication that will be necessary for the exchange of information between the FTDi and the computer.
- **C)** Initializes the LoRa Radio.



Image 5: LoraWan source code. Source: the author.

#### 2.2 Channels, regions and classes

In the source code "lorawan\_setup.c" we can configure the channels, regions and classes according to the needs of the microcontroller and the region where it is located. Each region of the planet or country has an operation frequency homologated for LoRa. In Brazil, the standard frequency is **902Mhz - 928Mhz**. In the image below the regions are highlighted.



Image 6 - Source code for channels and regions. Source: The author.

## **2.3 Device classes**

Also in the same source code file "**lorawan\_setup.c**", we can see (image 7) where the "classes" configuration function is. The LoRa specification works with three types of **classes**: Class A, Class B and Class C. Normally, all LoRaWAN devices must implement **Class A**, while Class B and Class C are extensions of the specification for **Class A** devices.

- Class A Bi-directional communication between devices such as sensors and microcontroller with the server. Fully optimized to reduce power consumption. One Rx1 window is opened when a message is sent.
- Class B Multiple Rx1 receive windows open. However, at set times.
- Class C Only two receiver windows are opened.



Image 7: Class source code. Source: The author.

## 2.3 Application Session Key

The library "lorawandefines.h", which is located inside the "includes" folder, is intended to implement the cryptographic element that supports LoRaWAN versions 1.0.x and 1.1.x. The code uses version 1.1.x.

1 *lorawandefines	i.h 🛛			90
13 #endif				00
14@/*!				
15 *******	***********************************		**********	
17 ******	*******	WARNING	*****	
18 The cr	vpto-element implementation	supports both 1.0.x and 1.	1.x LoRaWAN	
19 versio	ns of the specification.	and the second second second second		
20 Thus i	t has been decided to use t	he 1.1.x keys and EUI name	definitions.	
21 The be	low table shows the names e	quivalence between versions	1	
22	+	11 ×		
25	+======================================	1.1.X		
25	LORAWAN DEVICE EUI	LORAWAN DEVICE EUI		
26	+	++		
27	LORAWAN_APP_EUI	LORAWAN_JOIN_EUI		
28	+			
29	N/A	LORAWAN_APP_KET		
31	LORAWAN APP KEY	LORAWAN NWK KEY		
32	+	++		
33	LORAWAN_NWK_S_KEY	LORAWAN_F_NWK_S_INT_KEY		
34	+	++		
35	LORAWAN_NWK_S_KEY	LORAWAN_S_NWK_S_INT_KEY		
37	LORAWAN NWK S KEY	LORAWAN NWK S ENC KEY		
38	+	++		
39	LORAWAN_APP_S_KEY	LORAWAN_APP_S_KEY		
40	+	++		
41 ******	***************************************	*************	**********	
42 *******	**************************	************************	*****	
45				
45				
469 /*!				

Image 8: LoraWan library source code. Source: The author.

The LoRaWan protocol, for security, transmits data (payload) encrypted using the 128-bit AES algorithm with a key known as the "**Application Session Key**". Take a look at image 9:

- A) Represents the device (microcontroller) address to the network, works like a "MAC Address" of the device.
- B) Used for encryption and decryption of a payload.
- C) Normally used to interact between the device and the Network Server.

™ *lorawandefines.h 🖾	
64 * IEEE Organizationally Unique Identifier ( OU	I) (big endian)
66 */	1281101
67 #define IEEE_OUI	0x01, 0x01, 0x01
68 69® /*1	
70 * Mote device IEEE EUI (big endian)	
71 *	
73 */	
74 75 #define LORAWAN_DEVICE_EUI	{0xDE,0xAD,0xBE,0xEF,0xBE,0xEF,0xDE,0xAD}
76 77 #endif	
78 // { IEEE_OUI, 0x01, 0x01, 0x01, 0x01, 0x01 }	
79	
81 * App/Join server IEEE EUI (big endian)	
82 */	[ 0-01 0-01 0-01 0-01 0-01 0-01 0-00
84	{ 8,81, 8,81, 8,81, 8,81, 8,81, 8,81, 8,81, 8,81, 8,88
850/*!	B
80 Application Poot key	
88 #define LORAWAN_APP_KEY	{ 0x11, 0x1.
90°/*I	
91 * Network root key 92 * WARNING: FOR 1.0 x DEVICES IT IS THE \ref LO	
93 */	
94 #define LORAWAN_NWK_KEY	{ 0x11, 0x1
96 #if( OVER_THE_AIR_ACTIVATION == 0 )	
97	

Image 9: Session keys source code. Source: The author.

Finally, still in the same source code of the "**lorawandefines.h**" library, (**A**) we set the data transmission cycle "15000 ms" and (**B**) we set the **Class** type, as visualized in image 10 is **class A**.

lorawandefines.h 🕮 🔪	
147 */	
148 #ifndef APP_TX_DUTYCYCLE	
149 #define APP_TX_DUTYCYCLE 15000	
150 <b>#end1</b> †	
151e /*!	
152 * LoRaWAN Adaptive Data Rate	
153 * @note Please note that when ADR is enabled the end-device should be sta	.ic
154 */	
155 #define LORAWAN_ADR_STATE LORAWAN_ADR_ON	
156@/*!	
157 * LoRaWAN Default data Rate Data Rate	
158 * @note Please note that BL is used only when ADR is disabled	
159 */	
160 #define LORAWAN_DEFAULT_DATA_RATE DR_4	
162 × L-DeWON emplication met	
165 Corawan application port	
164 - Emote do not use 224. It is reserved for certification	
165 #define LORAWAN APP PORT 2	
1689 /*!	
169 * LoRaWAN default endNode class port	
170 🛫	
171 Idefine LORAWAN DEFAULT CLASS CLASS A	
172	
1730/*!	
174 * LoRaWAN default confirm state	
175 */	
176 #define LORAWAN_DEFAULT_CONFIRM_MSG_STATE LORAWAN_UNCONFIRMED_MS	i .
177	
1788/*!	
179 * User application data buffer size	
180 */	
101 HAASIMA LODALIAN ADD DATA DIEE STTE	

Image 10: Class and data transmission cycle definition. Source: The author.

In the source code "gpio.c" we can analyze which GPIO is set to receive the *push button*. In this case, it is GPIO\_4.

lorawa	n_setup.c 🕜 gpio.c 🖾	
77	INF_INTO_EUROTETIG(0.100_EUG))	
78	GPTO InitStruct Pin - RADIO SWITCH ENARLE Pin-	
79	GPTO TAILSTALL MADE - GPTO MADE OUTPUT PP:	
80	GPTO InitStruct Pull = GPTO NOPULI:	
81	GPIO InitStruct.Speed = GPIO SPEED FRED VERY HIGH:	
82	HAL GPIO Init(RADIO SWITCH ENABLE Port, &GPIO InitStruct):	
83 }		
84		
85° v	bid IRQHandler_Config(void) {	
86	GPIO_InitTypeDef GPIO_InitStructure;	
87	EXTI_ConfigTypeDef EXTI_Config_InitStructure;	
88		
89	/* Enable GPIOA and GPIOB clock */	
90	HAL_RCC_GPIOB_CLK_ENABLE();	
91	HAL_RCC_GPIOA_CLK_ENABLE();	
92	/* confirme DD A min on input florting */	
93	/~ Contigure PB.4 pin as input floating -/	
94	GPT0_INITStructure.Mode = GPT0_MODE_INFOT,	
96	GPIO_INISCRUCTURE Pin = GPIO_NOVEC,	
97	HAL GPTO Init/GPTOR & GPTO Init/Structure):	
98		
99	EXTI Config InitStructure.Line = EXTI LINE PB4;	
100	EXTI_Config_InitStructure.Trigger = EXTI_TRIGGER_RISING_EDGE;	
101	EXTI_Config_InitStructure.Type = EXTI_TYPE_EDGE;	
102		
103	HAL_EXTI_SetConfigLine(&HEXTI_InitStructure, &EXTI_Config_InitStructure);	
104	HAL_EXTI_RegisterCallback(&HEXTI_InitStructure, HAL_EXTI_COMMON_CB_ID, (void (*)(uint32_t))RadioOnDioIrq);	
105	HAL_EXTI_Cmd(&HEXTI_InitStructure, ENABLE);	
106		
107	HAL_EXII_CIEAFYEnding(@HEXII_INITSTRUCTURE);	
108		
		P

Image 11: GPIO source code. Source: The author.

## 2.2 Build Project

After analyzing the main structure that will be used in this application, right-click on the main project folder and then click "**Build Project**". This operation will compile all the code and consequently will automatically build the binary of the code.

🔊 workspace - 🤆	New	> CON	Applications\PushButton_LoRaWA	N\Inc\lorawandefines.h - WiSE Studio	0		
File Edit :	Go Into	Ý	Vindow Help				
		~	✓ ▼ : 2 回 回 1 : 2 ▼ 0 ▼ V	· ♀ ▼ ♀ ▼   ₫			Quick A
<ul> <li>✓ I Project Exp</li> <li>✓ I HTLRBL</li> </ul>	Show In Alt+Shift+	w > N	C_LORADEFINES_H_				
> 🔊 Inclu	Copy Ctrl	+C B	UG debug_configs.h"				
> 😂 Debi	> Debt Paste Ctriv	ete					
> & Mide Source > *		***************************************	**************************************	*************************			
■ HTLF	Move	*	**********************	**************************************	*******		
	Rename	F2 P	to-element implementation of the specification	n supports both 1.0.x and 1	L.1.x LoRaWAN		
≣ sip_h	import	Ĩ	has been decided to use	the 1.1.x keys and EUI name	e definitions.		
> 🎜 HTLRBL	🗳 Export	01	w table shows the names of the	equivalence between version	15: -+		
	Ruild Project	_	1.0.x	1.1.x	I		
	Clean Project	-	LORAWAN_DEVICE_EU	I LORAWAN_DEVICE_EUI	-+		
	Refresh	F5			+		
	Close Project		t		-+		
	Close Unrelated Project		N/A	LORAWAN_APP_KEY	1		
	Build Configurations	>	LORAWAN_APP_KEY	LORAWAN_NWK_KEY	1		
	Build Targets	>	LORAWAN_NWK_S_KEY	LORAWAN_F_NWK_S_INT_KEY	1		
	Index	>	LORAWAN_NWK_S_KEY	LORAWAN_S_NWK_S_INT_KEY	1		
	Run As	>	LORAWAN_NWK_S_KEY	LORAWAN_NWK_S_ENC_KEY	-+		
	🌣 Debug As	>	LORAWAN APP S KEY	LIORAWAN APP S KEY	*		
	Profile As	,	+	+	+		
	Team	> ×	***************************************	***************************************	***************************************		
	Compare With	> *	******	***********	**********		
	Restore from Local History	T	Tasks				*
3	Kun C/C++ Code Analysis		0 others				
_	Configure	>	^	Pocourco Path	Location	Tupo	

Image 12: Build Project. Source: The author.

The "Console" message shows that the Build was finished and that no errors occurred. We can also see that a new folder called "Binaries" has been created.

	Project Run Windo	v Help	• A vize		Duisk A	
	R lorawan catura c	lorawandofings h 18	2 (% 2 LB)			
HTLRBL32L-PushButton LoRaWAN	16 *********	*************	WARNING *********************	********		
> et Binaries > pi⊓indudes > te Application > te Debug	17 *************** 18 The crypto 19 versions o 20 Thus it ha 21 The below	-element implementation f the specification. s been decided to use th table shows the names ed	supports both 1.0.x and 1. he 1.1.x keys and EUI name quivalence between versions		1	
> 🕰 Drivers	22	+	1 1 x 1			
Imiddlewares     Imiddlewares     HTLRBL32L-PushButton_LoRaWAN.cfg     Imiddlewares	24 25	LORAWAN_DEVICE_EUI	LORAWAN_DEVICE_EUI	•		
HILRBL32L-PushButton_LoKaWAN.elf.ctg HTLRBL32LId	26 27	LORAWAN_APP_EUI	LORAWAN_JOIN_EUI			
₿ sip_htlrbl32l.xml	28	+	LORAWAN APP KEY			
HTLRBL32L-PushButton_LoRaWAN_Bluetooth	30	L LODAWAN ADD VEV				
	32 33	LORAWAN_NWK_S_KEY	LORAWAN_F_NWK_S_INT_KEY	-		
	34	LORAWAN_NWK_S_KEY	LORAWAN_S_NWK_S_INT_KEY			
	37 38	LORAWAN_NWK_S_KEY	LORAWAN_NWK_S_ENC_KEY	-		
	Problems Tasks	Console 22 Properties			⊕ <b>6</b> [2] 11 11 = 04   2	
	arm-none-eabi-size text data 90352 372 Finished building Finished building	12e e -B "HTLRBL32L-PushButt bss dec hext 12856 103580 1949c H : elf-size : HTLRBL32L-PushButton_L	ton_LoRaWAN.elf" filename HTLRBL32L-PushButton_LoRaWA LoRaWAN.hex	AN.elf		

Imagem 13: Arquivo Binário criado. Source: the author.

# **Additional reading**

GIT. **Git.** 2022. Available at: < <u>https://git-scm.com/</u>>. Accessed on july 28th 2022.

GIT-HUB. **Git guides.** 2022. Available at: < <u>https://github.com/git-guides</u> >. Accessed on july 28th 2022.

LORA-ALLIANCE. About-lorawan. Available at: < <u>https://lora-alliance.org/about-lorawan/</u> >. Accessed on july 28th 2022.

PREDICTABLEDESIGNS. Introduction to Embedded Firmware Development,2021. Available at: < <a href="https://predictabledesigns.com/introduction-to-embedded-firmware-development/">https://predictabledesigns.com/introduction-to-embedded-firmware-development/</a> >. Accessed on july 28th 2022.

THETHINGSNETWORK. **Security 2022.** Available at: < <u>https://www.thethingsnetwork.org/docs/lorawan/security/</u> >. Accessed on july 28th 2022.