

**GigaDevice Semiconductor Inc.**

**Migration from GD32E103/C103 series to  
GD32F30x series**

**Application Note  
AN047**

## Table of Contents

Table of Contents.....	2
List of Figures .....	3
List of Tables .....	4
1. Introduction.....	5
2. Pin compatibility .....	6
3. Internal resource compatibility.....	7
4. Program transplantation .....	9
4.1. IDE setting.....	9
4.2. FMC peripheral file replacement.....	9
4.3. Clock system configuration.....	10
5. Peripheral difference .....	11
5.1. PMU .....	11
5.2. CAN .....	11
5.3. EXMC.....	12
5.4. USB/USBFS .....	12
6. Precautions for transplantation.....	13
6.1. Software delay adjustment .....	13
6.2. Flash programme .....	13
6.3. Slow start-up time .....	14
7. Revision history.....	15

## List of Figures

Figure 4-1. Select chip model .....	9
Figure 4-2. Modify head file.....	10
Figure 4-3. Remove Flash wait cycle configuration .....	10
Figure 5-1. High-Driver mode.....	11

## List of Tables

Table 2-1. Pin Definitions Comparison .....	6
Table 3-1. Resources comparison .....	7
Table 6-1. GD32F30x Flash memory characteristics .....	13
Table 6-2. GD32E103/C103xx Flash memory characteristics .....	13
Table 6-3. GD32F30x series start-up timing .....	14
Table 6-4. GD32E103/C103xx series start-up timing .....	14
Table 7-1. Revision history .....	15

## 1. Introduction

This application note is designed to help you quickly migrate applications from GD32E103/C103 series MCU to GD32F30x series MCU.

In order to make better use of the information in this application note, you need to download it from the website [www.GD32MCU.com](http://www.GD32MCU.com), such as datasheet, user manual, official code and various development tools.

## 2. Pin compatibility

GD32F30x and GD32E103/C103xx series are pin to pin compatible under the same package. However, GD32E103xx has no CAN peripherals, so there are slight functional differences on the pins. Here, take GD32F303/F305/F307xx as an example.

[Table 2-1. Pin Definitions](#) shows the difference between GD32F303xx series and GD32F305/F307xx, GD32E103/C103xx series pins.

**Table 2-1. Pin Definitions Comparison**

Pin NO.	GD32F303xx	GD32F305/F307xx	GD32E103/C103xx
PF0	Default: PF0 Alternate: EXMC_A0 Remap: CTC_SYNC	Default: PF0 Alternate: EXMC_A0 Remap: CTC_SYNC	GD32E103/C103xx does not support 144Pin,so there is no PF0 pin
PB5	Default: PB5 Alternate: I2C0_SMBA, SPI2_MOSI, I2S2_SD Remap: TIMER2_CH1, SPI0_MOSI	Default: PB5 Alternate: I2C0_SMBA, SPI2_MOSI, I2S2_SD Remap: TIMER2_CH1, SPI0_MOSI, CAN1_RX	Default: PB5 Alternate: I2C0_SMBA, SPI2_MOSI, I2S2_SD Remap: TIMER2_CH1, SPI0_MOSI, CAN1_RX
PB6	Default: PB6 Alternate: I2C0_SCL, TIMER3_CH0 Remap: USART0_TX, SPI0_IO2	Default: PB6 Alternate: I2C0_SCL, TIMER3_CH0 Remap: USART0_TX, CAN1_TX, SPI0_IO2	Default: PB6 Alternate: I2C0_SCL, TIMER3_CH0 Remap: USART0_TX, CAN1_TX, SPI0_IO2
PB12	Default: PB12 Alternate: SPI1_NSS, I2C1_SMBA, USART2_CK, TIMER0_BRKIN, I2S1_WS	Default: PB12 Alternate: SPI1_NSS, I2C1_SMBA, USART2_CK, TIMER0_BKIN, I2S1_WS, CAN1_RX	Default: PB12 Alternate: SPI1_NSS, I2S1_WS, I2C1_SMBA, USART2_CK, TIMER0_BRKIN, CAN1_RX
PB13	Default: PB13 Alternate: SPI1_SCK, USART2_CTS, TIMER0_CH0_ON, I2S1_CK	Default: PB13 Alternate: SPI1_SCK, USART2_CTS, TIMER0_CH0_ON, I2S1_CK, CAN1_TX	Default: PB13 Alternate: SPI1_SCK, I2S1_CK, USART2_CTS, TIMER0_CH0_ON, CAN1_TX, I2C1_TXFRAME

**Note:**

- Both GD32F305xx and GD32F307xx have two CAN interfaces, and GD32F303xx has only one CAN interface.
- The TIMER8\_CH0/ TIMER8\_CH1/ TMIER12\_CH0 channels of GD32E103 are available in GD32F30xVG/I/K devices.

### 3. Internal resource compatibility

[Table 3-1. Resources comparison](#) shows that the resource comparison between GD32F30x and GD32E103/C103xx (taking the comparison of GD32F303xx, GD32F305xx and GD32E103/C103xx as an example).

**Table 3-1. Resources comparison**

Internal resource	GD32F303xx	GD32F305xx	GD32E103/C103xx	Compatibility description
Frequency	108MHz	120MHz	120MHz	Frequency is different
core	M4 core	M4 core	M4 core	Fully compatible
Power supply range	2.6V-3.6V	2.6V-3.6V	1.8V-3.6V	The power supply range of 30x series is relatively narrow
Flash	256-3072KB	128-1024KB	64-128K	There are differences in memory operation modes
RAM	48-96KB	64-96K	32K	
GPTM	4/10	4/10	4/10	Compatible with peripherals existing in the corresponding chip
Advanced TM	1/2	1/2	1/2	Compatible with peripherals existing in the corresponding chip
Basic TM	2	2	2	Fully compatible
Systick	1	1	1	Fully compatible
Watch dog	2	2	2	Fully compatible
RTC	1	1	1	Fully compatible
U(S)ART	3/5	5	2/3/5	Compatible with peripherals existing in the corresponding chip
I2C	2	2	1/2	Compatible with peripherals existing in the corresponding chip
SPI/IIS	3/2	3/2	1-3/2	Compatible with peripherals existing in the corresponding chip
CAN	1	2	0/2	GD32F303/F305xx only support CAN2.0
USBFS	1(USBD)	1(USBOTG)	1(USBOTG)	USBD belongs to GD32F303/F305xx and GD32E103/C103xx compatible

## Migration from GD32E103/C103 series to GD32F30x series

Internal resource	GD32F303xx	GD32F305xx	GD32E103/C103xx	Compatibility description
GPIO	37/51/80/112	51/80/112	26/37/51/80	Compatible with peripherals existing in the corresponding chip
EXMC	0/1	0/1	0/1	The GD32F305xx and GD32E103/C103xx in 100pin package has an exmc, and some functions of exmc are added to GD32F305xx
ADC(CH)	3(16/21)	2(16)	2(10/16)	Compatible with peripherals existing in the corresponding chip
DAC	2	2	2	Fully compatible
CTC	1	1	1	Fully compatible



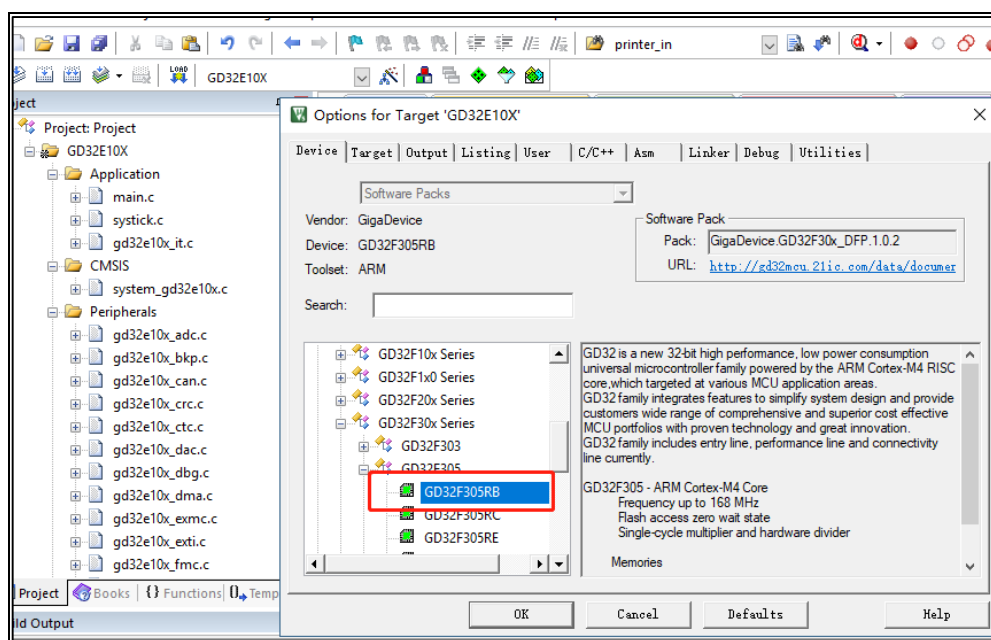
## 4. Program transplantation

GD32E103/C103xx peripherals and clocks are basically compatible with GD32F30x interconnection series. Some modifications and adjustments are also required when migrating applications based on GD32E103/C103xx firmware library to GD32F30x.

### 4.1. IDE setting

1. When using the MDK environment, install the GD32F30x series plug-in and replace it with the corresponding model of GD32F30x (here take porting to GD32F305RB as an example).

**Figure 4-1. Select chip model**



2. When using IAR environment, the equipment model is also changed in the project.

### 4.2. FMC peripheral file replacement

GD32E103/C103xx adopts embedded e-flash architecture. The flash programming of GD32E103/C103xx supports whole word and half word programming;

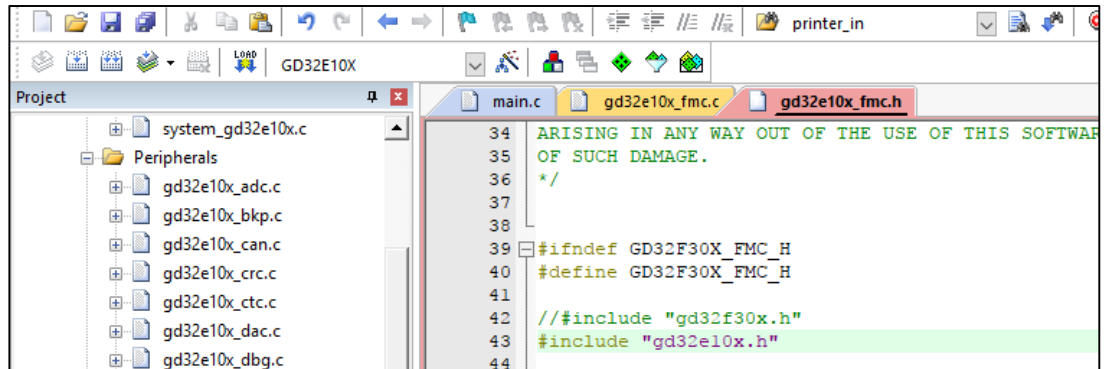
The SIP overlay architecture adopted by GD32F30x also supports whole word and half word programming in flash programming, and is divided into bank0 and bank1. The operation of option bytes also needs to be programmed according to words.

It is suggested to adopt a convenient migration method: replace the code in `gd32e10x_fmc.c` and `gd32e10x_fmc.h` in the original project with the code in `gd32f30x_fmc.c` and `gd32f30x_fmc.h` in GD32F30x firmware library. After replacement, you need to change all

## Migration from GD32E103/C103 series to GD32F30x series

relevant `#include "gd32e10x.h"` to `#include "gd32f30x.h"`, as shown in [Figure 4-2. Modify head file](#). GD32E103xx firmware library can be obtained from the official website or online disk.

**Figure 4-2. Modify head file**



```

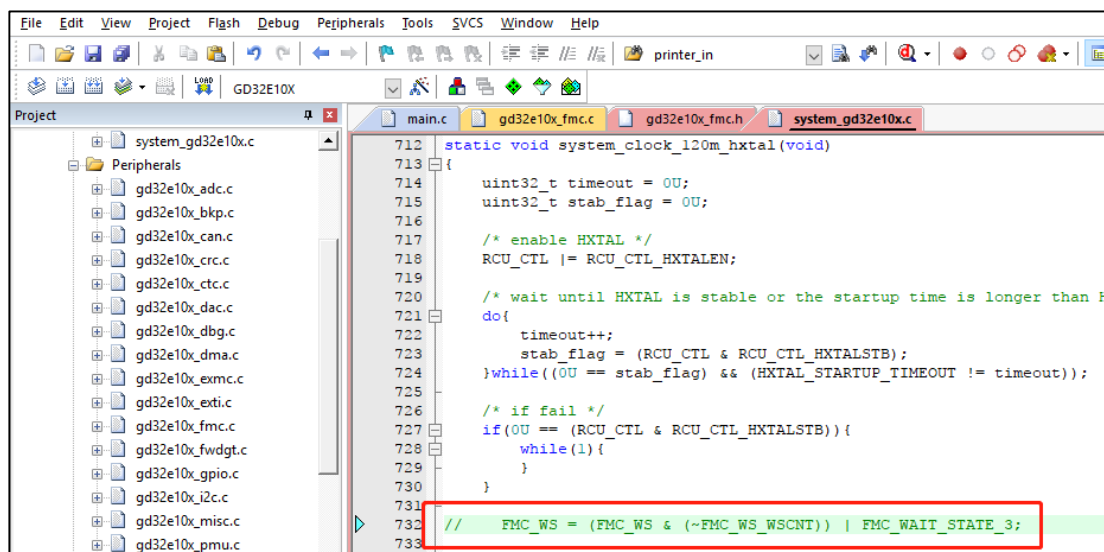
34 ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE
35 OF SUCH DAMAGE.
36 */
37
38
39 #ifndef GD32F30X_FMC_H
40 #define GD32F30X_FMC_H
41
42 // #include "gd32f30x.h"
43 #include "gd32e10x.h"
44
  
```

### 4.3. Clock system configuration

The clock configuration process of GD32F30x interconnected series is the same as that of GD32E103/C103xx series. There are flash waiting cycles on GD32E103/C103xx, and corresponding flash waiting cycles need to be added before clock configuration. However, F30x series is only divided into code area and data area, and code area is zero waiting area, so there is no need to insert any waiting cycle into flash

Remove the configuration of flash waiting cycle from the clock configuration function in the `system_gd32f30x.c` file, as shown in [Figure 4-3. Remove Flash wait cycle configuration](#).

**Figure 4-3. Remove Flash wait cycle configuration**



```

712 static void system_clock_120m_hxtal(void)
713 {
714     uint32_t timeout = 0U;
715     uint32_t stab_flag = 0U;
716
717     /* enable HXTAL */
718     RCU_CTL |= RCU_CTL_HXTALEN;
719
720     /* wait until HXTAL is stable or the startup time is longer than
721     do{
722         timeout++;
723         stab_flag = (RCU_CTL & RCU_CTL_HXTALSTB);
724     }while((0U == stab_flag) && (HXTAL_STARTUP_TIMEOUT != timeout));
725
726     /* if fail */
727     if(0U == (RCU_CTL & RCU_CTL_HXTALSTB)){
728         while(1){
729             }
730     }
731
732     // FMC_WS = (FMC_WS & (~FMC_WS_WSCNT)) | FMC_WAIT_STATE_3;
733
  
```

**Note:** If GD32F303xx is used for replacement, it is recommended to replace the `system_gd32e10x.c` file. Refer to the replacement method of FMC for the operation mode.

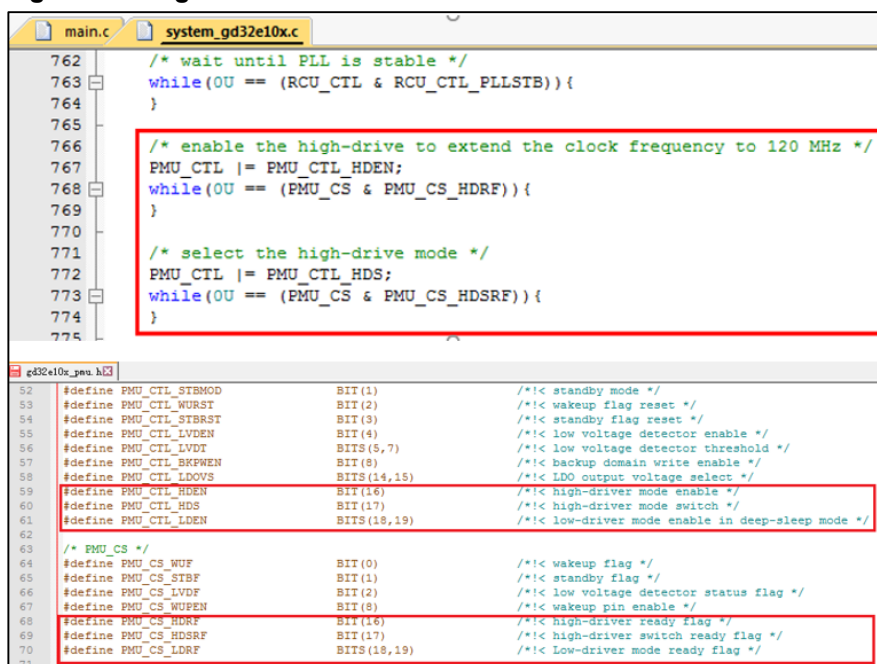
## 5. Peripheral difference

GD32F30x and GD32E103/C103xx are mostly compatible with peripherals.

### 5.1. PMU

GD32F30x PMU Module adds high drive mode, which needs to be manually defined if necessary. As shown in [Figure 5-1. High-Driver mode](#).

**Figure 5-1. High-Driver mode**



```

main.c | system_gd32e10x.c
762      /* wait until PLL is stable */
763      while(OU == (RCU_CTL & RCU_CTL_PLLSTB)){
764      }
765
766      /* enable the high-drive to extend the clock frequency to 120 MHz */
767      PMU_CTL |= PMU_CTL_HDEN;
768      while(OU == (PMU_CS & PMU_CS_HDRF)){
769      }
770
771      /* select the high-drive mode */
772      PMU_CTL |= PMU_CTL_HDS;
773      while(OU == (PMU_CS & PMU_CS_HDSRF)){
774      }
775
gd32e10x_pmu.h
52 #define PMU_CTL_STEMOD BIT(1)          /*< standby mode */
53 #define PMU_CTL_WURST BIT(2)          /*< wakeup flag reset */
54 #define PMU_CTL_STBRST BIT(3)        /*< standby flag reset */
55 #define PMU_CTL_LVDEN BIT(4)         /*< low voltage detector enable */
56 #define PMU_CTL_LVDT  BITS(5,7)      /*< low voltage detector threshold */
57 #define PMU_CTL_BKPWEN BIT(8)        /*< backup domain write enable */
58 #define PMU_CTL_LDOVS  BITS(14,15)   /*< LDO output voltage select */
59 #define PMU_CTL_HDEN  BIT(16)        /*< high-driver mode enable */
60 #define PMU_CTL_HDS   BIT(17)        /*< high-driver mode switch */
61 #define PMU_CTL_LDEN  BITS(18,19)    /*< low-driver mode enable in deep-sleep mode */
62
63 /* PMU_CS */
64 #define PMU_CS_WUF    BIT(0)         /*< wakeup flag */
65 #define PMU_CS_SIBF  BIT(1)         /*< standby flag */
66 #define PMU_CS_LVDF  BIT(2)         /*< low voltage detector status flag */
67 #define PMU_CS_WUPEN BIT(8)         /*< wakeup pin enable */
68 #define PMU_CS_HDRF  BIT(16)        /*< high-driver ready flag */
69 #define PMU_CS_HDSRF BIT(17)        /*< high-driver switch ready flag */
70 #define PMU_CS_LDRF  BITS(18,19)    /*< Low-driver mode ready flag */
71

```

In addition, the power consumption of GD32F30x is relatively higher than that of GD32E103/C103xx in the same mode. For details, please refer to the data manual.

### 5.2. CAN

1. The CAN of GD32F30x is ordinary CAN2.0, the maximum communication baud rate is 1mbit / s. The CAN of GD32C103xx is CAN FD. There are 14 filters in non GD32F30x CL (interconnected) series products; There are 28 filters in GD32F30x CL (interconnected) series products;
2. The firmware driver of CAN shall be modified by referring to the transplantation mode of FMC above.

**Note:** GD32F303xx is a 512 byte dedicated SRAM shared by USBD and CAN for data buffering, so it cannot be used at the same time.

### **5.3. EXMC**

The EXMC of GD32F30x has 8-bit / 16 bit NAND flash controller and 16 bit PC card controller more than GD32E103/C103xx series, and each bank has an independent signal.

### **5.4. USB/USBFS**

The USBFS of GD32E103xx and GD32F305/F307xx are basically the same. The USB of GD32F303xx is USB and shares 512 bytes of special SRAM with can for data buffering. It cannot be used simultaneously with can interface. The USB of GD32F305/F307xx is USB OTG. They have separate USB SRAM and can be used simultaneously with CAN.

## 6. Precautions for transplantation

### 6.1. Software delay adjustment

The flash operation of GD32E103xx has a waiting period. The flash of GD32F30x series is designed with zero waiting, so the efficiency of GD32F30x will be slightly higher than that of GD32E103/C103xx at the same main frequency. If the user code uses for loop or while loop to delay, the delay time will be shorter on GD32F30x series. It is necessary to appropriately increase the delay parameter or use timer as the delay function. Special attention should be paid in the application scenario of using GPIO to simulate the communication protocol.

### 6.2. Flash programme

The memory erasing and programming time of GD32F30x is longer than that of GD32E103xx, as shown in [Table 6-1. GD32F30x Flash memory characteristics](#) and [Table 6-2. GD32E103/C103xx Flash memory characteristics](#).

**Table 6-1. GD32F30x Flash memory characteristics**

Symbol	Parameter	Conditions	Min <sup>(1)</sup>	Typ <sup>(1)</sup>	Max <sup>(2)</sup>	Unit
PE <sub>CYC</sub>	Number of guaranteed program /erase cycles before failure (Endurance)	T <sub>A</sub> = -40 °C ~ +85 °C	100	—	—	kcycles
t <sub>RET</sub>	Data retention time	—	—	20	—	years
t <sub>PROG</sub>	Word programming time	T <sub>A</sub> = -40°C ~ +85 °C	—	37.5	86	μs
t <sub>ERASE</sub>	Page erase time	T <sub>A</sub> = -40°C ~ +85 °C	—	45	200/300 <sup>(3)</sup>	ms
t <sub>MERASE(256K)</sub>	Mass erase time	T <sub>A</sub> = -40°C ~ +85 °C	—	1	4.8/8.0 <sup>(4)</sup>	s
t <sub>MERASE(512K)</sub>	Mass erase time	T <sub>A</sub> = -40°C ~ +85 °C	—	4	19.2/32 <sup>(5)</sup>	s
t <sub>MERASE(1MB)</sub>	Mass erase time	T <sub>A</sub> = -40°C ~ +85 °C	—	6	28.8/48 <sup>(6)</sup>	s
t <sub>MERASE(2MB)</sub>	Mass erase time	T <sub>A</sub> = -40°C ~ +85 °C	—	10	48/80 <sup>(7)</sup>	s
t <sub>MERASE(3MB)</sub>	Mass erase time	T <sub>A</sub> = -40°C ~ +85 °C	—	14	67.2/112 <sup>(8)</sup>	s

(1) Based on characterization, not tested in production.

(2) Guaranteed by design, not tested in production.

(3) Max value with <50K cycles is 200 ms and >50K & <100K cycles is 300 ms.

(4) Max value with <50K cycles is 4.8 s and >50K & <100K cycles is 8.0 s.

(5) Max value with <50K cycles is 19.2 s and >50K & <100K cycles is 32 s.

(6) Max value with <50K cycles is 28.8 s and >50K & <100K cycles is 48 s.

(7) Max value with <50K cycles is 48 s and >50K & <100K cycles is 80 s.

(8) Max value with <50K cycles is 67.2 s and >50K & <100K cycles is 112 s.

**Table 6-2. GD32E103/C103xx Flash memory characteristics**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
PE <sub>CYC</sub> <sup>(1)</sup>	Number of guaranteed program /erase cycles before failure(Endurance)	T <sub>A</sub> = -40 °C ~ +85 °C	100	—	—	kcycles

## Migration from GD32E103/C103 series to GD32F30x series

$t_{RET}^{(1)}$	Data retention time	10k cycles at $T_A = 85\text{ }^\circ\text{C}$	10	—	—	years
$t_{PROG}^{(2)}$	Word <sup>(3)</sup> programming time	$T_A = -40\text{ }^\circ\text{C} \sim +85\text{ }^\circ\text{C}$	37	—	44	$\mu\text{s}$
$t_{ERASE}^{(2)}$	Page erase time	$T_A = -40\text{ }^\circ\text{C} \sim +85\text{ }^\circ\text{C}$	3.2	—	4	ms
$t_{MERASE}^{(2)}$	Mass erase time	$T_A = -40\text{ }^\circ\text{C} \sim +85\text{ }^\circ\text{C}$	8	—	10	ms

(1) Based on characterization, not tested in production.

(2) Guaranteed by design, not tested in production.

(3) Word is 32 bits or 64 bits depend on PGW bit in FMC\_WS register.

### 6.3. Slow start-up time

The power on start time of GD32F30x is longer than that of GD32E103/C103xx. The specific contents are shown in [Table 6-3. GD32F30x series start-up timing](#) and [Table 6-4. GD32E103/C103xx series start-up timing](#).

**Table 6-3. GD32F30x series start-up timing**

Symbol	Parameter	Conditions	Typ	Unit
$t_{start-up}$	Start-up time	Clock source from HXTAL	144	ms
		Clock source from IRC8M	144	

(1) Based on characterization, not tested in production.

(2) After power-up, the start-up time is the time between the rising edge of NRST high and the main function.

(3) PLL is off.

**Table 6-4. GD32E103/C103xx series start-up timing**

Symbol	Parameter	Conditions	Typ	Unit
$t_{start-up}$	Start-up time	Clock source from HXTAL	468	$\mu\text{s}$
		Clock source from IRC8M	86.8	

(1) Based on characterization, not tested in production.

(2) After power-up, the start-up time is the time between the rising edge of NRST high and the first I/O instruction conversion in SystemInit function.

(3) PLL is off.

## 7. Revision history

Table 7-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Mar.15 2022

## Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company under the intellectual property laws and treaties of the People's Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as its suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.