Readme

APM32E030 SDK

Rev：V1.0

1 Introduction

The Geehy Semiconductor APM32E030 TINY board software development kit includes a series driver library, a group of example applications that demonstrate key peripheral functionality, and other development files.

Software development kit have a hierarchy as follows:

- SDK directory
  - Boards
  - Documents
  - Examples
  - Libraries
  - Middlewares
  - Package

Table of Contents

2 About boards

The boards folder includes a board support package for APM32E030 TINY board. It can help drive the peripheral circuit or components on the board quickly. The BSP can be found in the Boards directory.

The BSP provided are built for APM32E030 TINY board compatibility. For other user development board use, some minor modifications may be required.

Boards have a hierarchy as follows:

- Boards folder
  - Board_APM32E030_TINY
    - inc
    - src
  - Board.c
  - Board.h

Board APM32E030 TINY include following board support package:

- Board_APM32E030_TINY src folder
  - Board_APM32E030_TINY

3 About documents

The documents folder includes a link file that can be redirected to the technical support center of Geehy semiconductor. The document can be found in the Documents directory.

4 About examples

The example applications can be found in the Examples directory.

The examples provided are built for APM32E030 TINY board compatibility. For other user development board use, some minor modifications may be required.

Example projects have a hierarchy as follows:

- Example folder
  - Include
  - Project
    - IAR
    - Eclipse
    - MDK
  - Source

All example applications tested with: **APM32E030 StdPeriphDriver v1.0.0**, include the following examples:

- Examples
  - ADC
    - ADC_AnalogWindowWatchdog
    - ADC_ContinuousConversion
    - ADC_TMRTrigger
    - ADC_MultiChannelScan
  - CRC
    - CRC_CalcMessage
  - DMA
    - DMA_ADC
    - DMA_Usart
    - DMA_MemoryToMemory
  - EINT
    - EINT

* FMC
    - FMC_Write
    - FMC_WriteOB
* GPIO
    - GPIO_Toggle
* I2C
    - I2C_TwoBoards
    - I2C_TwoBoards_DMA
* IAP
    - IAP_Application1
    - IAP_Application2
    - IAP_BootLoader
* IWDT
    - IWDT_FeedDog
    - IWDT_FeedDog_Window
* NVIC
    - NVIC_WFI
* PMU
    - PMU_WakeUp
* RCM
    - RCM_ClockSwitch
* RTC
    - RTC_Alarm
    - RTC_Calendar
    - RTC_Stamp
    - RTC_TimeStamp
    - RTC_LPWR_Wakeup
* RTOS

- FreeRTOS
- RT-Thread
- RTX

* SPI
  - SPI_FullDuplex
  - SPI_TwoBoards
  - SPI_TwoBoards_DMA
* SysTick
  - SysTick
* Template
  - Template
* TMR
  - TMR_6Steps
  - TMR_32BitCount
  - TMR_ComplementaryOutput
  - TMR_DMABurst
  - TMR_EncoderInterface
  - TMR_InputCapture
  - TMR_OCActive
  - TMR_OCInactive
  - TMR_OCToggle
  - TMR_ParallelSynchro
  - TMR_PWMOutput
  - TMR_SinglePulse
  - TMR_SynchronizationWithTMR1
  - TMR_TimeBase
  - TMR_DMA
* USART

4.1 ADC_AnalogWindowWatchdog

4.1.1 Example Description

This example describes how to use ADC1 to monitor the voltage of ADC1_Channel0 continuously. When input Voltage of ADC1_Channel0(PA0) voltage is lower than 1.33v， LED2 is on, When input Voltage of ADC1_Channel0(PA0) voltage is higher than 1.66v, LED3 is on.

The converted voltage is displayed on serial assistant through USART1

4.1.2 Directory content

This example can be found in the ADC_AnalogWindowWatchdog directory.

4.2 ADC_ContinuousConversion

4.2.1 Example Description

This example describes how to use the ADC1 to convert continuously the voltage

applied to the APM32E030 TINY ADC1_Channel0 input.

The converted voltage is displayed on serial assistant through USART1.

4.2.2 Directory content

This example can be found in the ADC_ContinuousConversion directory.

4.3 ADC_TMRTrigger

4.3.1 Example Description

This example describes how to use the ADC1 to convert continuously the voltage

applied to the APM32E030 TINY ADC1_Channel0 input.

Each time the TMR1 CC4 event occurs, the ADC convert the variable voltage.

The converted voltage is displayed on serial assistant through USART1.

4.3.2 Directory content

This example can be found in the ADC_TMRTrigger directory.

4.4 ADC_MultiChannelScan

4.4.1 Example Description

This example describes how to use the ADC1 to scan continuously the voltage applied to ADC_Channel0 and ADC_Channel1 and ADC_Channel2 input.

The converted voltage is displayed on serial assistant through USART1.

4.4.2 Directory content

This example can be found in the ADC_MultiChannelScan directory.

4.5 CRC_CalcMessage

4.5.1 Example Description

Write the calculated data to CRC DATA register and get the calculated result. The computer of serial debugging assistant can display the corresponding information for Computed CRC.

4.5.2 Directory content

This example can be found in the CRC_CalcMessage directory.

4.6 DMA_ADC

4.6.1 Example Description

This example provides an example of how to use a DMA channel to transfer continuously a data from a peripheral (ADC1) to DMA transfer. The ADC channel1 for APM32E030 TINY Board is configured to be converted when device startup.

4.6.2 Directory content

This example can be found in the DMA_ADC directory.

4.7 DMA_Usart

4.7.1 Example Description

This example provides a basic communication between USART1 and USART2 using DMA1 capability.

After system reset, USART2 Transmit data from DMA_USART_TxBuf to USART1. Data received by USART1 is transferred by DMA and stored in DMA_USART_RxBuf. If the data of DMA_USART_TxBuf and DMA_USART_RxBuf are the same, LED2 and LED3 will constantly flicker alternately, or flicker together.

4.7.2 Directory content

This example can be found in the DMA_Usart directory.

4.8 DMA_MemoryToMemory

4.8.1 Example Description

This example shows how to configure the DMA peripheral to transmit data from memory to memory. After system reset, data transmit form one group to another through DMA. If the data received is equal to the data send, LED2 will light, otherwise, LED3 will light.

4.8.2 Directory content

This example can be found in the DMA_ MemoryToMemory directory.

4.9 EINT

4.9.1 Example Description

This example shows how to configure external interrupt lines. In this example, 2 EINT lines (KEY1、KEY2) when using the APM32E030 TINY BOARD are configured to generate an interrupt on each falling edge. In the interrupt routine a led connected to a specific GPIO pin is toggled.

In this example

- EINT0 is mapped to PA0(KEY2)
- EINT1 is mapped to PA1(KEY1)

After EINT configuration：

- when falling edge is detected on EINT0, LED2 toggles
- when falling edge is detected on EINT1, LED3 toggles

4.9.2 Directory content

This example can be found in the EINT directory.

4.10 FMC_Write

4.10.1 Example Description

This example provides a description of how to program the flash address of APM32E030.

After Reset, the Flash will be unlocked. Then erase the specifies address and write a data in the address. In the end, lock the flash. The data of the address after erasing and programing will be displayed on serial assistant through USART1.

4.10.2 Directory content

This example can be found in the FMC_Write directory.

4.11 FMC_WriteOB

4.11.1 Example Description

This example provides a description of how to program the OB address.

After Reset, the Flash will be unlocked. Then erase the OB and write data in user data. In the end, lock the flash. The data of the address after erasing and programing will be displayed on serial assistant through USART1.

4.11.2 Directory content

This example can be found in the FMC_WriteOB directory.

4.12 GPIO_Toggle

4.12.1 Example Description

This example describes how to use DOUT for toggling IO. The IO of LED2 and LED3 is configured to toggle constantly. The phenomenon of LED2 and LED3 constantly flickered alternately.

4.12.2 Directory content

This example can be found in the GPIO_Toggle directory.

4.13 I2C_TwoBoards

4.13.1 Example Description

This example shows how to control I2C devices and communicate between two different boards.

- At startup, Boards master and slave are both in slave receiver mode and wait for messages to be received.

- When KEY1 is pressed on Board master, the master transmitter sent "Hello master" to Board slave. Message is displayed on serial assistant through USART2.

- When KEY1 is pressed on Board slave, the slave transmitter sent "Hello master" to Board master. The message is displayed on serial assistant through USART2.

4.13.2 Directory content

This example can be found in the I2C_TwoBoards directory.

4.14 I2C_TwoBoards_DMA

4.14.1 Example Description

This example shows how to control I2C devices and communicate by DMA between two different boards.

- At startup, Boards master and slave are both in slave receiver mode and wait for messages to be received.

- When KEY1 is pressed on Board master, the master transmitter sent "Hello master" to Board slave. Message is displayed on serial assistant through USART2.

- When KEY1 is pressed on Board slave, the slave transmitter sent "Hello master" to Board master. The message is displayed on serial assistant through USART2.

4.14.2 Directory content

This example can be found in the I2C_TwoBoards_DMA directory.

4.15 IAP_Application1

4.15.1 Example Description

This example shows how to generate a APP firmware to IAP. LED2 are toggled with a timing defined by the Delay function.

4.15.2 Directory contents

This example can be found in the IAP_Application1 directory.

4.16 IAP_Application2

4.16.1 Example Description

This example shows how to generate a APP firmware to IAP.LED3 are toggled with a timing defined by the Delay function.

4.16.2 Directory contents

This example can be found in the IAP_Application2 directory.

4.17 IAP_BootLoader

4.17.1 Example Description

The example aims to show how to configure a bootloader firmware to IAP.

When device connect to HyperTerminal right, a usart menu will show to user.

4.17.2 Directory contents

This example can be found in the IAP_ BootLoader directory.

4.18 IWDT_FeedDog

4.18.1 Example Description

The example aims to show how to configure IWDT and feed dog to prevent a system reset. After IWDT initialization, System enters into infinite loop and feed dog within one second to prevent system reset. In the loop, System will output information to serial assistant to display system status.

4.18.2 Directory contents

This example can be found in the IWDT_FeedDog directory.

4.19 IWDT_FeedDog_Window

4.19.1 Example Description

The example aims to show how to update the IWDG reload counter at special window period. After IWDT initialization, System enters into infinite loop and resets when feeding dog not in window period. In the same time, System will output information to serial assistant to display system status.

The IWDG counter is refreshed in the eint interrupt when press the KEY1 in window value to prevent a IWDG reset.

4.19.2 Directory content

This example can be found in the IWDT_FeedDog_Window directory

4.20 NVIC_WFI

4.20.1 Example Description

This example describes how to use WFI event to enter sleep mode and wake up using external interrupt.

At startup, press KEY2(PA0) to occur wait for Interrupt(WFI) event, and device will enter sleep mode. The device will wake up if press KEY2 again.

4.20.2 Directory content

This example can be found in the NVIC_WFI directory.

4.21 PMU_WakeUp

4.21.1 Example Description

This example shows how to enter the system by external interrupt to:

- STANDBY mode and wake-up from this mode either with the RESET or give PC13 a falling edge to recover.

- SLEEP mode and wake-up from this mode either with the RESET or give PA6 a falling edge to recover.

- STOP mode and wake-up from this mode with the RESET.

phenomenon:

-If press KEY1(SLEEP mode): LED2 is on, while LED3 remain in the previous state. After giving PA6 a falling edge, system running from reset state.

-If press KET2(STANDBY mode): LED2 and LED3 are off. After giving PC13 a rising edge, system continue running.

-If low down PA7(STOP mode): LED2 and LED3 remain in the previous state. After reset, system recover to normal state.

4.21.2 Directory content

This example can be found in the PMU_WakeUp directory.

4.22 RCM_ClockSwitch

4.22.1 Example Description

This example shows how to:

- Configure the PLL (clocked by HSE) as System clock source

- Configure HSI as System clock source

- Configure HSE as System clock source

- Output the System clock on MCO pin.

Through two buttons(KEY1、KEY2), switch the system clock.

phenomenon:

- If press KEY1(PLL as clock source): LED2 flashing quickly

- If press KEY2(HSI as clock source): LED2 flashing normally

- You can monitor the system clock on MCO pin (PA.8) or on serial assistant.

4.22.2 Directory content

This example can be found in the RCM_ClockSwitch directory.

4.23 RTC_Alarm

4.23.1 Example Description

This example shows how to configure RTC and ALARM.

phenomenon:

- After initialization, Alarm begin to count down with LED2 is on. Five second later, Alarm is waking up and LED2 is off.

- You can monitor the system state on serial assistant.

4.23.2 Directory content

This example can be found in the RTC_Alarm directory.

4.24 RTC_Calendar

4.24.1 Example Description

This example shows how to:

- Using RTC to set the system time and date.

- recover system time from reset state by using RTC_WriteBackup and RTC_ReadBackup function.

phenomenon:

- Time and date information are displayed through serial assistant.

- If reset (Power is on), system will read data from the backup area and display it.

- If reset (System Power from off to on), system will reinitialize the RTC.

4.24.2 Directory content

This example can be found in the RTC_Calendar directory.

4.25 RTC_Stamp

4.25.1 Example Description

This example shows how to write/read data to/from RTC Backup data registers and demonstrates the Tamper detection feature.

The associated firmware performs the following:

1. It configures the Tamper pin to be falling edge, and enables the Tamper interrupt.

2. It writes the data to all RTC Backup data registers

3. On applying a low level on the RTC_AF1 pin (PC.13, connected to Tamper button), the RTC backup data registers are reset and the Tamper interrupt is generated.

phenomenon:

- If Tamper1 interrupt happen, LED2 is on
- If Tamper2 interrupt happen, LED3 is on
- Time and Backup data are displayed in serial assistant.

4.25.2 Directory content

This example can be found in the RTC_Stamp directory.

4.26 RTC_TimeStamp

4.26.1 Example Description

This example provides a short description of how to use the RTC peripheral and the Time Stamp feature.

The associated firmware performs the following:

1.It configures the RTC TimeStamp pin to be falling edge and enables the TimeStamp detection.

2.When RTC_TimeStamp pin is on the air, a interrupt will be generated and print TimeStamp information to serial assistant.

phenomenon:

- TimeStamp information is displayed in serial assistant.

4.26.2 Directory content

This example can be found in the RTC_TimeStamp directory.

4.27 RTC_LPWR_Wakeup

4.27.1 Example Description

This example shows how to configure RTC and ALARM .

phenomenon :

- After initialization, Alarm begin to count down with LED2 is on. Five second later, Alarm is waking up and LED2 is off.

- You can monitor the system state on serial assistant.

4.27.2 Directory content

This example can be found in the RTC_LPWR_Wakeup directory.

4.28 FreeRTOS

4.28.1 Example Description

This example describes show how to how to use FreeRTOS create multiple tasks.

Usart test task: USART1 and USART2 send or received data to each other. Verification will occur after transmission,

if send and receive data pass, LED3 will be on all the time.

if send and receive data fault, LED3 will be off all the time.

if send or received data fault, LED3 will be constantly flickered alternately.

Led toggle task: The IO of LED2 is configured to toggle constantly

The phenomenon of LED2 constantly flickered alternately,

if Data transmission pass, LED3 will be on, and data interaction process can be displayed using serial assistant.

4.28.2 Directory content

This example can be found in the FreeRTOS directory.

4.29 RT-Thread

4.29.1 Example Description

This example describes how to use RT-Thread for APM32E030.

The IO of LED2 and LED3 is configured to toggle constantly.

The phenomenon of LED2 and LED3 constantly flickered alternately.

4.29.2 Directory content

This example can be found in the RT-Thread directory.

4.30 RTX

4.30.1 Example Description

This example describes show how to how to use RTX5 create multiple tasks.

Usart test task: USART1 and USART2 send or received data to each other. Verification will occur after transmission, if send and receive data pass, LED3 will be on all the time. if send and receive data fault, LED3 will be off all the time. if send or received data fault, LED3 will be constantly flickered alternately.

Led toggle task: The IO of LED2 is configured to toggle constantly

The phenomenon of LED2 constantly flickered alternately, if Data transmission pass, LED3 will be on, and data interaction process can be displayed using serial assistant.

4.30.2 Directory content

This example can be found in the RTX directory.

4.31 SPI_FullDuplex

4.31.1 Example Description

This demo is based on the APM32E030 board, it shows how to use SPI peripheral. By making a master/slave full duplex communication between the SPI and the UART1.

The phenomenon of serial assistant can display information from USART2.

4.31.2 Directory content

This example can be found in the SPI_FullDuplex directory.

4.32 SPI_TwoBoards

4.32.1 Example Description

This example provides a small application in which system sends and receives data by polling though using SPI firmware library. All received information will be displayed by serial assistant.

To use this example, you need to load it on two APM32E030 boards (let's call them Board master and Board slave). Then connect these two boards through SPI lines and must master and slave connect to the same GND. When compare buffer is the same, the LED2 is on.

The phenomenon of data interaction process can be displayed using serial assistant.

4.32.2 Directory content

This example can be found in the SPI_TwoBoards directory.

4.33 SPI_TwoBoards_DMA

4.33.1 Example Description

This example provides a small application in which system sends and receives data by DMA though using SPI firmware library. All received information will be displayed by serial assistant.

To use this example, you need to load it on two APM32E030 boards (let's call them Board master and Board slave). Then connect these two boards through SPI lines and must master and slave connect to the same GND. When compare buffer is the same, the LED2 is on.

The phenomenon of data interaction process can be displayed using serial assistant.

4.33.2 Directory content

This example can be found in the SPI_TwoBoards_DMA directory.

4.34 SysTick

4.34.1 Example Description

This example shows how to configure the SysTick to generate a time base equal to 1 ms. The system clock is set to 8 MHz, the SysTick is clocked by the HSE clock.

A "Delay" function is implemented based on the SysTick end-of-count event.

For LEDs are toggled with a timing defined by the Delay function.

4.34.2 Directory content

This example can be found in the SysTick directory.

4.35 Template

4.35.1 Example Description

This demo is based on the APM32E030 TINY board. it provides a template project.

4.35.2 Directory contents

This example can be found in the Template directory.

4.36 TMR_6Steps

4.36.1 Example Description

The program to show how to configure the TMR1 peripheral to generate 6 Steps.

In this example, a software COM event is generated each 100 ms.

The TMR1 is configured in Timing Mode, each time a COM event occurs, a new TMR1 configuration will be set in advance.

4.36.2 Directory contents

This example can be found in the TMR_6Steps directory.

4.37 TMR_32BitCount

4.37.1 Example Description

This example describes how to configure the TMR3 and TMR15 realize the 32-bit timer.

TMR15 as High 16 bit count value, TMR3 as Low 16 bit count value. User can view the counter value through serial terminal.

4.37.2 Directory contents

This example can be found in the TMR_32BitCount directory.

4.38 TMR_ComplementaryOutput

4.38.1 Example Description

This example shows how to configure the TMR1 peripheral to generate complementary TMR1 signals, to insert a defined dead time value.

The phenomenon of observe whether the output waveform is correct through the oscilloscope. Besides LED2 and LED3 constantly flashing.

using TMR1 CH1 PA7 to output PWM

using TMR1 CH1N PA8 to output PWM

4.38.2 Directory contents

This example can be found in the TMR_ComplementaryOutput directory.

4.39 TMR_DMABurst

4.39.1 Example Description

The program to show how to configure the TMR1 channel period and the duty cycle by DMA burst to generate 7 PWM with 7 different duty cycles (80%, 70%, 60%, 50%, 40%, 30% and 20%).

On the DMA update request, the DMA will do 6 transfers of half words into TMR1 registers (AUTORLD, REPCNT, CC1, CC2, CC3, CC4).

4.39.2 Directory contents

This example can be found in the TMR_DMABurst directory.

4.40 TMR_EncoderInterface

4.40.1 Example Description

This example describes how to configure the TMR1 peripheral to Encoder mode.

4.40.2 Directory contents

This example can be found in the TMR_EncoderInterface directory.

4.41 TMR_InputCapture

4.41.1 Example Description

This example shows how to configure the TMR1 peripheral to capture the internal clock source from pin MCO.

The result will be displayed on serial assistant through USART2.

4.41.2 Directory contents

This example can be found in the TMR_InputCapture directory.

4.42 TMR_OCActive

4.42.1 Example Description

The program to show how to configure the TMR3 peripheral to generate 4 different signals with four different delays.

The CHx delay correspond to the time difference between PE6 falling edge and TMR3_CHx signal rising edges. Reset system and display TMR3 waveform by oscilloscope.

4.42.2 Directory contents

This example can be found in the TMR_OCActive directory.

4.43 TMR_OCInactive

4.43.1 Example Description

The program to show how to configure the TMR3 peripheral in Output Compare Inactive mode.

4.43.2 Directory contents

This example can be found in the TMR_OCInactive directory.

4.44 TMR_OCToggle

4.44.1 Example Description

The program to show how to configure the TMR3 peripheral to generate 4 waveform with 4 different frequencies.

4.44.2 Directory contents

This example can be found in the TMR_OCToggle directory.

4.45 TMR_ParallelSynchro

4.45.1 Example Description

This example shows how to synchronize TMR peripherals in parallel mode.

4.45.2 Directory contents

This example can be found in the TMR_ParallelSynchro directory.

4.46 TMR_PWMInput

4.46.1 Example Description

This example describes how to use TMR3 Channel_2 (PA7) measure frequency and duty cycle of external sign.

4.46.2 Directory contents

This example can be found in the TMR_PWMInput directory.

4.47 TMR_PWMOutput

4.47.1 Example Description

This example shows how to configure the TMR1 peripheral to generate PWM signals with different duty cycles. The TMR1 waveform can be displayed using an oscilloscope.

using TMR1 CHANNEL1(PA8) to output PWM

4.47.2 Directory contents

This example can be found in the TMR_PWMOutput directory.

4.48 TMR_SinglePulse

4.48.1 Example Description

This example shows how to use the TMR peripheral to generate a One-pulse Mode after a falling edge of an external signal is received in Timer Input pin.

The TMR3 waveform can be displayed using an oscilloscope.

4.48.2 Directory content

This example can be found in the TMR_SinglePulse directory..

4.49 TMR_SynchronizationWithTMR1

4.49.1 Example Description

This example shows how to synchronize TMR peripherals in cascade mode, two timers TMR1 and TMR3 are used.

The phenomenon of observe whether the output waveform is correct through the oscilloscope.

Using TMR3 CH1(PA6) to output PWM

4.49.2 Directory content

This example can be found in the TMR_SynchronizationWithTMR1 directory.

4.50 TMR_TimeBase

4.50.1 Example Description

This example aims to show how to realize timing one second by using TMR14 peripheral generating time base.

LED2 and LED3 will toggle per second.

4.50.2 Directory content

This example can be found in the TMR_TimeBase directory.

4.51 TMR_DMA

4.51.1 Example Description

The program to show how to use DMA to transfer Data from memory to TMR1 Capture Compare Register1 to change the Duty Cycle.

4.51.2 Directory content

This example can be found in the TMR_DMA directory.

4.52 USART_Interrupt

4.52.1 Example Description

Through The computer of serial debugging assistant, display the message sent and received between the MCU and USART1.

The phenomenon of serial assistant can display information from USART1.

4.52.2 Directory content

This example can be found in the USART_Interrupt directory.

4.53 USART_Polling

4.53.1 Example Description

The program aims to show how to send or received data by using USART, in this case, USART1 and USART2 send or received data to each other. Verification will occur after transmission, if passed, LED2 will be on.

The phenomenon of data interaction process can be displayed using serial assistant.

4.53.2 Directory content

This example can be found in the USART_Polling directory.

4.54 WWDT_OverTime

4.54.1 Example Description

This example aims to show how to use WWDT.

If is_OverTime = 0, System would not reset for feeding dog timely. LED2 Toggle.

If is_OverTime = 1, System will reset. LED3 ON.

4.54.2 Directory content

This example can be found in the WWDT_OverTime directory.

5 About libraries

The libraries folder includes a series library. It can provide supports for APM32E030 MCU such as device support and standard peripheral. The libraries can be found in the Libraries directory.

APM32E030 MCU include following library:

- Libraries folder
  - APM32E030_StdPeriphDriver
  - CMSIS
  - Device

6 About middlewares

The middlewares can be found in the Middlewares directory.

The middlewares used by APM32E030 TINY include following:

- Middlewares folder

7 About Package

The Package folder includes Geehy APM32E030_DFP Package. The Package can be found in the Package directory.

The Package used by APM32E030 TINY include following:

- Package folder
  - * Geehy.APM32E030_DFP.1.0.0.pack

8 Revision History

Table 1 File Revision History

Date	Rev	Description
2024.07.31	1.0	First release version of APM32E030 SDK V1.0

Statement

This document is formulated and published by Geehy Semiconductor Co., Ltd. (hereinafter referred to as “Geehy”). The contents in this document are protected by laws and regulations of trademark, copyright and software copyright. Geehy reserves the right to make corrections and modifications to this document at any time. Read this document carefully before using Geehy products. Once you use the Geehy product, it means that you (hereinafter referred to as the “users”) have known and accepted all the contents of this document. Users shall use the Geehy product in accordance with relevant laws and regulations and the requirements of this document.

1. Ownership

This document can only be used in connection with the corresponding chip products or software products provided by Geehy. Without the prior permission of Geehy, no unit or individual may copy, transcribe, modify, edit or disseminate all or part of the contents of this document for any reason or in any form.

The “极海” or “Geehy” words or graphics with “®” or “TM” in this document are trademarks of Geehy. Other product or service names displayed on Geehy products are the property of their respective owners.

2. No Intellectual Property License

Geehy owns all rights, ownership and intellectual property rights involved in this document.

Geehy shall not be deemed to grant the license or right of any intellectual property to users explicitly or implicitly due to the sale or distribution of Geehy products or this document.

If any third party’s products, services or intellectual property are involved in this document, it shall not be deemed that Geehy authorizes users to use the aforesaid third party’s products, services or intellectual property. Any information regarding the application of the product, Geehy hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party, unless otherwise agreed in sales order or sales contract.

3. Version Update

Users can obtain the latest document of the corresponding models when ordering Geehy products.

If the contents in this document are inconsistent with Geehy products, the agreement in the sales order or the sales contract shall prevail.

4. Information Reliability

The relevant data in this document are obtained from batch test by Geehy Laboratory or cooperative third-party testing organization. However, clerical errors in correction or errors caused by differences in testing environment may occur inevitably. Therefore, users should understand that Geehy does not bear any responsibility for such errors that may occur in this document. The relevant data in this document are only used to guide users as performance parameter reference and do not constitute Geehy's guarantee for any product performance.

Users shall select appropriate Geehy products according to their own needs, and effectively verify and test the applicability of Geehy products to confirm that Geehy products meet their own needs, corresponding standards, safety or other reliability requirements. If losses are caused to users due to user's failure to fully verify and test Geehy products, Geehy will not bear any responsibility.

5. Legality

USERS SHALL ABIDE BY ALL APPLICABLE LOCAL LAWS AND REGULATIONS WHEN USING THIS DOCUMENT AND THE MATCHING GEEHY PRODUCTS. USERS SHALL UNDERSTAND THAT THE PRODUCTS MAY BE RESTRICTED BY THE EXPORT, RE-EXPORT OR OTHER LAWS OF THE COUNTRIES OF THE PRODUCTS SUPPLIERS, GEEHY, GEEHY DISTRIBUTORS AND USERS. USERS (ON BEHALF OR ITSELF, SUBSIDIARIES AND AFFILIATED ENTERPRISES) SHALL AGREE AND PROMISE TO ABIDE BY ALL APPLICABLE LAWS AND REGULATIONS ON THE EXPORT AND RE-EXPORT OF GEEHY PRODUCTS AND/OR TECHNOLOGIES AND DIRECT PRODUCTS.

6. Disclaimer of Warranty

THIS DOCUMENT IS PROVIDED BY GEEHY "AS IS" AND THERE IS NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

GEEHY'S PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED FOR USE AS CRITICAL COMPONENTS IN MILITARY, LIFE-SUPPORT, POLLUTION CONTROL, OR HAZARDOUS SUBSTANCES MANAGEMENT SYSTEMS, NOR WHERE FAILURE COULD RESULT IN INJURY, DEATH, PROPERTY OR ENVIRONMENTAL DAMAGE.

IF THE PRODUCT IS NOT LABELED AS "AUTOMOTIVE GRADE," IT SHOULD NOT BE CONSIDERED SUITABLE FOR AUTOMOTIVE APPLICATIONS. GEEHY ASSUMES NO LIABILITY FOR THE USE BEYOND ITS SPECIFICATIONS OR GUIDELINES.

THE USER SHOULD ENSURE THAT THE APPLICATION OF THE PRODUCTS COMPLIES WITH ALL RELEVANT STANDARDS, INCLUDING BUT NOT LIMITED TO SAFETY, INFORMATION SECURITY, AND ENVIRONMENTAL REQUIREMENTS. THE USER ASSUMES FULL RESPONSIBILITY FOR THE SELECTION AND USE OF GEEHY PRODUCTS. GEEHY WILL BEAR NO RESPONSIBILITY FOR ANY DISPUTES ARISING FROM THE SUBSEQUENT DESIGN OR USE BY USERS.

7. Limitation of Liability

IN NO EVENT, UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL GEEHY OR ANY OTHER PARTY WHO PROVIDES THE DOCUMENT AND PRODUCTS "AS IS", BE LIABLE FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE DOCUMENT AND PRODUCTS (INCLUDING BUT NOT LIMITED TO LOSSES OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY USERS OR THIRD PARTIES). THIS COVERS POTENTIAL DAMAGES TO PERSONAL SAFETY, PROPERTY, OR THE ENVIRONMENT, FOR WHICH GEEHY WILL NOT BE RESPONSIBLE.

8. Scope of Application

The information in this document replaces the information provided in all previous versions of the document.

Geehy Semiconductor Co.,Ltd.

Bldg.1, No.83 Guangwan Street, Zhuhai, Guangdong, China   +86 0756 6299999   www.geehy.com