DTP3 User's API Guide

DAINCUBE Corp.

ARM Cortex-M4 STM32F429IGT6

FORM 141021F – 2015.11.30

DTP3 User's API Guide
Form 141021F-151130— Novermber, 2015

㈜Daincube
Web: www.daincube.com
E-mail: sales@daincube.com
Tel: 82-32-329-9783~4
Fax: 82-32-329-9785

#401-701, Bucheon TechnoPark 4-Danji,
655 Pyeongcheon-ro, Wonmi-gu, Bucheon-Si,
Gyeonggi-Do, Republic of Korea

Preface

Important information
This documentation is intended for qualified audience only. The product described herein is not an end user product. It was developed and manufactured for further processing by trained personnel.

Disclaimer
Although this document has been generated with the utmost care no warranty or liability for correctness or suitability for any particular purpose is implied. The information in this document is provided "as is" and is subject to change without notice.

Trademarks
All used product names, logos or trademarks are property of their respective owners

Product support
DAINCUBE Corp.
Web: www.daincube.com
E - MAIL: support@daincube.com

Safety precautions

Be sure to observe all of the following safety precautions.

Strict observance of these warning and caution indications are a MUST for preventing accidents, which could result in bodily injury and substantial property damage. Make sure you fully understand all definitions of these terms and related symbols given below, before you proceed to the manual.

Symbols

The following symbols may be used in this specification:

Warning : Warnings indicate conditions that, if not observed, can cause personal injury.

Caution : Cautions warn the user about how to prevent damage to hardware or loss of data.

Note : Notes call attention to important information that should be observed.

REVISON HISTORY

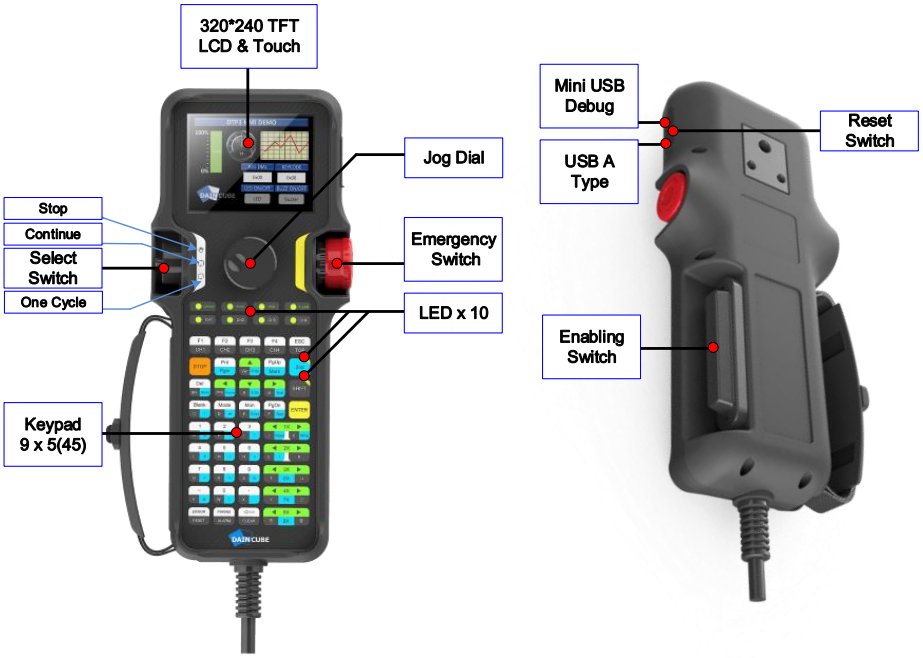
Revision	Data	Comment
Version 1.0	2014.10.13	Initial Version
Version 2.0	2015.07.25	Board revision
Version 2.1	2015.09.14	Modify keypad code value, additional UART XXX_Put_Len()
Version 2.2	2016.05.11	Fix typing error, remove debug port
Version 2.3	2016.06.07	Remove RS-232 port, switch function
Version 2.4	2017.03.16	Manual revision
Version 2.5	2017.07.10	Additional description of RS-485 communication
Version 2.6	2017.12.05	Fix external font typing error

Contents

1. Introduction

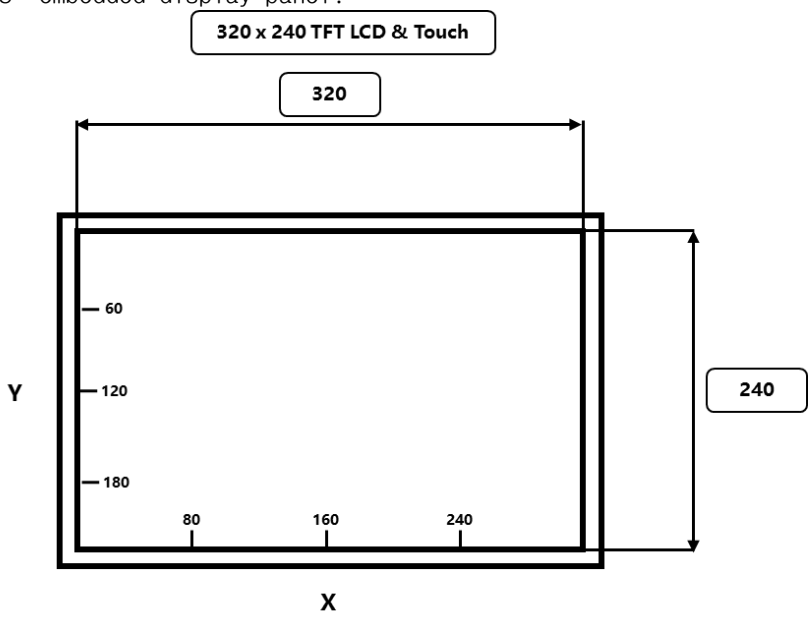
This document is for applying of DTP3(Daincube Teach Pendant Cortex_M4) to software development.
Daincube corporation provides DTP3_DEVKIT for software engineer.

DTP3 Platform and Specification
OS is Window7_32bit, development tool is Atollic's software TrueSTUDIO and debugging and Hex file download by ST-Link/V2 JTAG.



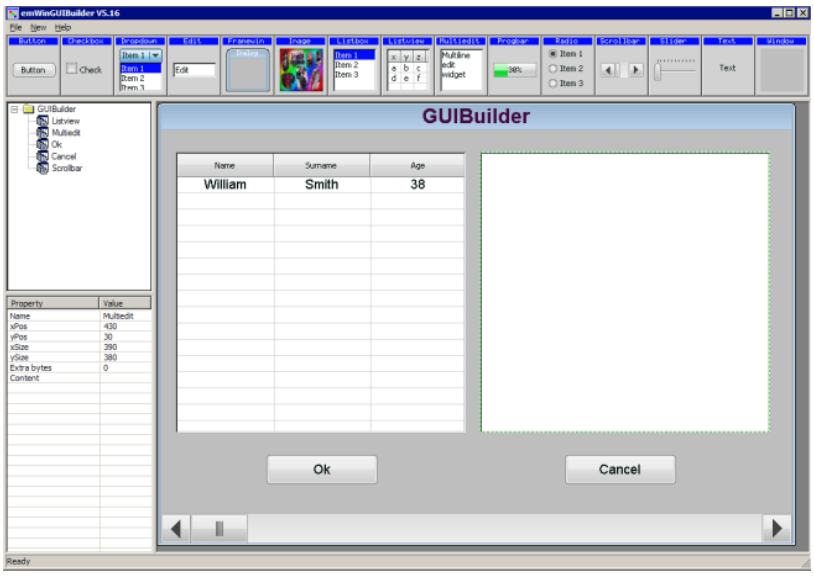
2. How to use the Display & Touch

DTP3 is installed 3.5' embedded display panel.



2.1. How to use the GUIBuilder.exe

DTP3 GUI development tool is "GUIBuilder.exe". This development tool is supported by SEGGER Company. (File Path : DTP3_DEVKITW02_DTP3_SWW03_SoftwareWSTemWin)
"GUIBuilder.exe" make Widgets, dialog boxes, checkboxes, buttons. If you makes UI form, this tool makes .c file.
"STemWin5.pdf" describe more detail functions.



2.2. Display API

Routine	Description
Daincube Display API	
void Display_Init(void)	- Initialize Display, Touch Driver and ready to use GUI.
STemWin GUI API (reference to STemWin5.pdf API)	
Int GUI_Init(void)	- Initialize internal data in emWin library.
void GUI_Clear(void)	- Clear Window widget.
Int GUI_Exec(void)	- Refresh Window widget.

2.2.1. void Display_Init(void)

- Description

  This function is Initialize Display, Touch, SDRAM and ready to use STemWin library.

- Header

  Display.h

2.2.2. Int GUI_Init(void)

- Description

  This function is Initialize internal data in emWin library.

- Return Value

  Int(data type) : Return to initialization result.

value	Meaning
0	- Initialization success.
Non zero value	- Initialization fail.

- Header

  GUI.h

2.2.3. void GUI_Clear(void)

- Description

  This function is clear window widget.

- Header

  GUI.h

2.2.4. Int GUI_Exec(void)

- Description

  This function is refresh window widget.

- Return Value

  Int(data type) : Return to value of re-drawing result.

value	Meaning
0	- Fail.
1	- Success.

- Header

  GUI.h

2.3. Example

```
#include "stm32f4xx.h"
#include "WindowDLG.h"
#include "Display.h"


int main(void)
{
    Display_Init();

    CreateWindow();


    while(1) {
        GUI_Exec();
    }
}
```

3. How to use Keypad

Keypad 9 x 5(45EA)

3.1. Keypad API

Routine	Description
void KPD_Init(void)	- Initialize Keypad.
struct input_event key_code	- Check on state of Keypad and data value.

3.1.1. void KPD_Init(void)

- Description

  This function is ready to use Keypad. Keypad consist of interrupt method. If keypad pressed, occur interrupt and operate timer. Timer is periodically search Key code value and timer changes waiting mode.

- Header

  KPD.h

3.1.2. struct input_event key_code

- Description

  This function is check state of keypad. So you can handling keypad function.

- Struct Variable

  key_code.code : This chart is each keypad code. Each keypad code was defined.

KEY_F1	KEY_F2	KEY_F3	KEY_F4	KEY_ESC
KEY_S	KEY_G	KEY_UP	KEY_PAGEUP	KEY_N
KEY_DELETE	KEY_LEFT	KEY_DOWN	KEY_RIGHT	KEY_LEFTSHIFT
KEY_C	KEY_D	KEY_E	KEY_PAGEDOWN	KEY_ENTER
KEY_1	KEY_2	KEY_3	KEY_J	KEY_K
KEY_4	KEY_5	KEY_6	KEY_O	KEY_P
KEY_7	KEY_8	KEY_9	KEY_T	KEY_U
KEY_MINUS	KEY_0	KEY_DOT	KEY_Y	KEY_Z
KEY_F5	KEY_F6	KEY_BACKSPACE	KEY_F7	KEY_F8

< Keypad 9 x 5 >

key_code.value : This value is state of keypad.

value	Meaning
0	- Keypad Up
1	- Keypad Down

- Header

KPD.h

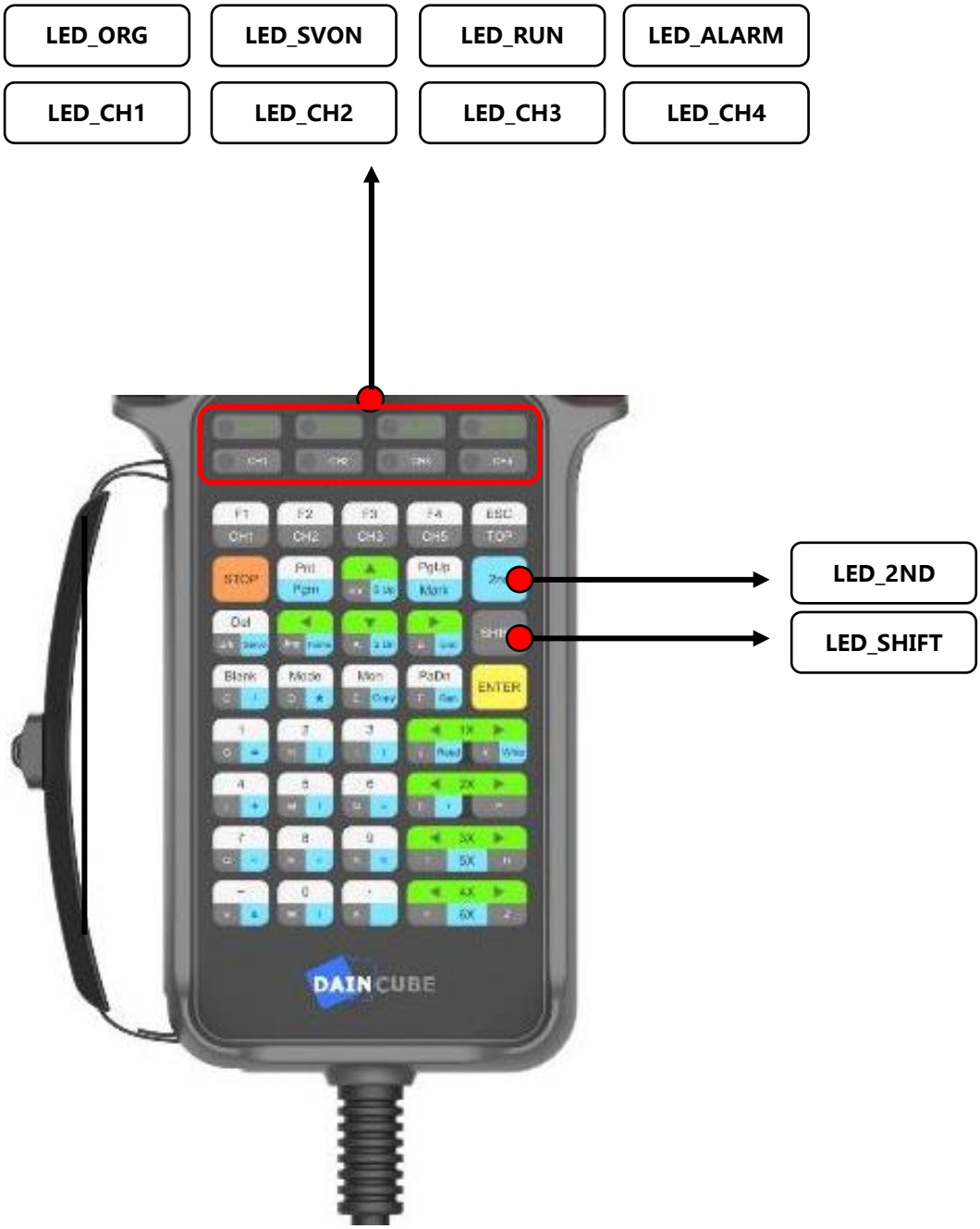
3.2. Example

```
#include "stm32f4xx.h"
#include "KPD.h"


int main(void)
{
    /*Keypad Driver initialization*/
    KPD_Init();

    while(1)
    {
        if(key_code.value == 1) {
            if(key_code.code == KEY_F1) {
                //
            }
        }
    }
}
```

4. How to use LED



4.1. LED API

Routine	Description
void ETC_Init(void)	- Initialize LED, Buzzer, Switch.
void LED_ALL_ON_OFF(char mod)	- All LED is Turn ON.
void LED_ON_OFF(char mod, char data)	- Each LED is Turn ON or OFF.

4.1.1. void ETC_Init(void)

- Description

  This function is Initialize LED, Buzzer, Switch. LED. LED controlled by 16bit I2C Driver.

- Header

  ETC.h

4.1.2. void LED_ALL_ON_OFF(char mod)

- Description

  This function is turn on or off all LED.

- Parameter

  mod : Select LED ON state or LED OFF state.

Value	Meaning
LED_ON	- Turn ON of all LED.
LED_OFF	- Turn OFF of all LED.

- Header

  ETC.h

4.1.3. void LED_ON_OFF(char mod, char data)

- Description

  This function is turn on or off each LED.

- Parameter

  mod : Select state of LED.

value	Meaning
LED_ON	- LED ON.
LED_OFF	- LED OFF.

  data : The data value of the LED to be controlled.

value	Meaning
LED_SHIFT	–  LED data
LED_2ND	–  LED data
LED_CH1	–  LED data
LED_CH2	–  LED data
LED_CH3	–  LED data
LED_CH4	–  LED data
LED_ORG	–  LED data
LED_SVON	–  LED data
LED_RUN	–  LED data
LED_ALARM	–  LED data

- Header

  ETC.h

4.2. Example

```
#include "stm32f4xx.h"
#include "ETC.h"
#include "Delay.h"


int main(void)
{
    /*ETC Driver initialization*/
    ETC_Init();
    Delay_Init();

    while(1)
    {
        LED_ALL_ON_OFF(LED_ON);
        Delay_ms(500);
        LED_ALL_ON_OFF(LED_OFF);
        Delay_ms(500);
        LED_ON_OFF(LED_ON, LED_CH1);
        Delay_ms(500);
        LED_ON_OFF(LED_OFF, LED_CH1);
        Delay_ms(500);
    }
}
```

5. How to use Buzzer

5.1. Buzzer API

Routine	Description
void ETC_Init(void)	- Initialize LED, Buzzer, Switch.
void BUZ_ON_OFF(char mod)	- Buzzer is Turn ON or OFF.

5.1.1. void ETC_Init(void)

- Description

  This function is initialize LED, Buzzer, Switch. Buzzer controlled by GPIO ON/OFF method.

- Header

  ETC.h

5.1.2. void BUZ_ON_OFF(char mod)

- Description

  This function is turn on or off Buzzer.

- Parameter

  mod : Select state of Buzzer.

value	Meaning
BUZ_ON	- Buzzer is Turn ON.
BUZ_OFF	- Buzzer is Turn OFF.

- Header

  ETC.h

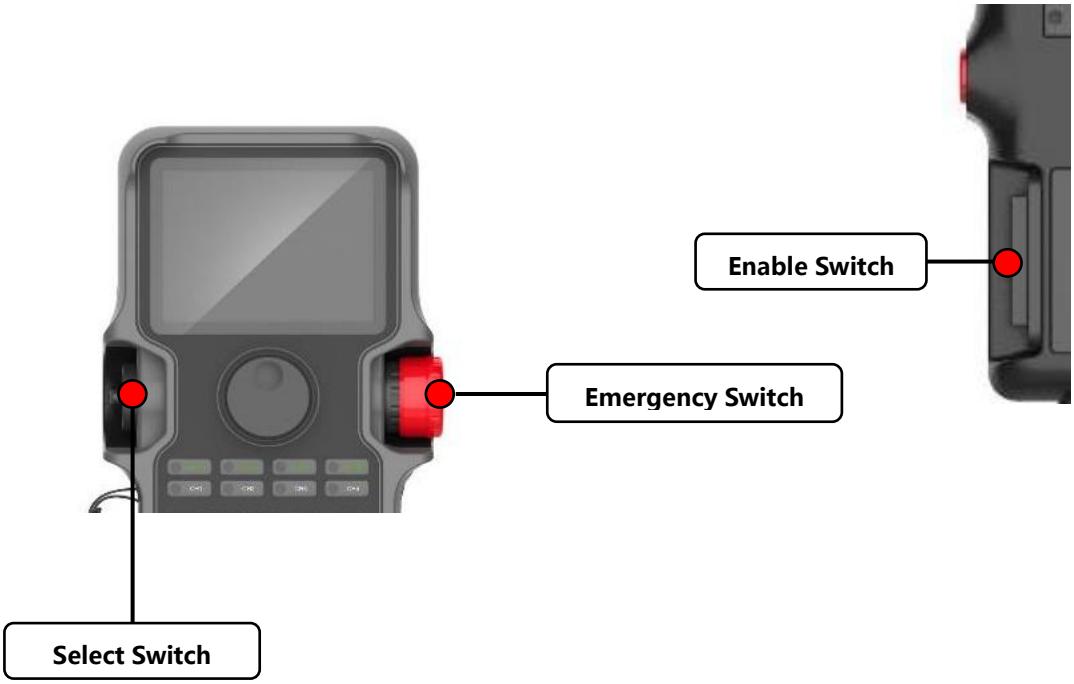
5.2. Example

```
#include "stm32f4xx.h"
#include "ETC.h"
#include "Delay.h"


int main(void)
{
    /*ETC Driver initialization*/
    ETC_Init();
    Delay_Init();

    while(1)
    {
        BUZ_ON_OFF(BUZ_ON);
        Delay_ms(500);
        BUZ_ON_OFF(BUZ_OFF);
        Delay_ms(500);
    }
}
```

6. How to use Switch



Switches are direct connection, then did not software processing.

7. How to use Serial

7.1. Serial API

Routine	Description
void RS485_Init(uint32_t baudrate)	- Initialize RS-485 Port.
void RS485_Puts(volatile char *s)	- Send data to RS-485 Port.
void RS485_Puts_Len(volatile char *s, char len)	- Send length of data to RS-485 Port.
void RS485_Direction(char mode)	- Select RS-485 direction.
volatile FIFO_TypeDef rs485_buffer	- Structure value of RS-485 receive buffer.
void Buffer_Init(volatile FIFO_TypeDef *buffer)	- Initialize RS-485 receive buffer.
char Buffer_Get(volatile FIFO_TypeDef *buffer, uint8_t *ch)	- Get 1byte data of RS-485 Port receive buffer.
char Buffer_Get_Len(volatile FIFO_TypeDef *buffer, uint8_t *ch, int len)	- Get length of data RS-485 Port receive buffer.
char Buffer_Is_Empty(volatile FIFO_TypeDef *buffer)	- Check empty RS-485 Port receive buffer.

7.1.1. void RS485_Init(uint32_t baudrate)

- Description

  This function is initialize RS-485 Port and ready to use TX, RX, Direction. RS-485 communication is always receive mode.

- Parameter

  Baudrate : Select speed for RS-485 port.

- Header

  UART.h

7.1.2. void RS485_Puts(volatile char *s)

- Description

  This function is Send data to RS-485 Port. Data type is string.

  (End of data is separated by a NULL value.)

- Parameter

  *s : Send address value of data for RS-485 port.

- Header

  UART.h

7.1.3. void RS485_Puts_Len(volatile char *s, char len)

- Description

  This function is send length of data to RS-485 Port.

- Parameter

  *s : Send address value of data for RS-485 port.

  len : Define length of data to RS-485 Port.

- Header

  UART.h

7.1.4. void RS485_Direction(char mode)

- Description

  This function is RS-485 Port is used TX, RX, Direction function. RS-485 communication is always receive mode.

- Parameter

  mod : Select direction of RS-485 communication.

value	Meaning
RS485_INPUT	- Receive mode
RS485_OUTPUT	- Send mode

- Header

  UART.h

7.1.5. volatile FIFO_TypeDef rs485_buffer

- Description

  This value is RS-485 receive buffer.

- Struct Variable

  rs485_buffer.count : Check on RS-485 receive buffer data.

- Header

  UART.h

7.1.6. void Buffer_Init(volatile FIFO_TypeDef *buffer)

- Description

  This function is initialize RS-485 receive buffer.

- Parameter

  *buffer : RS-485 buffer containing the data.

value	Meaning
rs485_buffer	- Receive buffer of RS-485

- Header

  UART.h

7.1.7. char Buffer_Get(volatile FIFO_TypeDef *buffer, uint8_t *ch)

- Description

  This value get 1byte data of RS-485 Port receive buffer.

- Parameter

  *buffer : RS-485 buffer containing the data.

value	Meaning
rs485_buffer	- Receive buffer of RS-485

  *ch : Send address of value which to store the data of 1byte.

- Return Value

  char : Return to reading count of byte.

value	Meaning
0	- None read data
1	- Read 1byte data

- Header

  UART.h

7.1.8. char Buffer_Get_Len(volatile FIFO_TypeDef *buffer, uint8_t *ch, int len)

- Description

  This value get data length with RS-485 Port receive buffer.

- Parameter

  *buffer : RS-485 buffer containing the data.

value	Meaning
rs485_buffer	- Receive buffer of RS-485

  *ch : Send address of value which to store the length of data.

  len : Define length of data.

- Return Value

  char : Return to length of data.

value	Meaning
0	- None data

N	- length of data

- Header

  UART.h

7.1.9. char Buffer_Is_Empty(volatile FIFO_TypeDef *buffer)

- Description

  This value is check RS-485 buffer state.

- Parameter

  *buffer : RS-485 buffer containing the data.

value	Meaning
rs485_buffer	- Receive buffer of RS-485

- Return Value

  char : Return to current buffer state.

value	Meaning
0	- None buffer data
1	- Exist buffer data

- Header

  UART.h

7.2. Example

7.2.1. RS485 Communication

The following example is an example using 10 protocols when communicating with DTP3 and PC.

```
#include "stm32f4xx.h"
#include "UART.h"


int main(void)
{
     uint8_t rec_buf[20] = {0, };

     RS485_Init(115200);  // initialize USART2 @ 115200 baud

     RS485_Direction(RS485_OUTPUT);
     RS485_Puts("₩n RS485 OK₩r₩n");
     RS485_Puts("/*****************************/₩r₩n");
     RS485_Puts("   DTP3-M Sample₩r₩n");
     RS485_Puts("   DAINCUBE₩r₩n");
     RS485_Puts("/*****************************/₩r₩n");
     RS485_Direction(RS485_INPUT);

     while(1) {
          if ( Buffer_Is_Empty(&rs485_buffer) == 0 ) {
               memset(rec_buf, 0, sizeof(rec_buf) );

               while(rs485_buffer.count >= 10 ) {
                    Buffer_Get_Len(&rs485_buffer, rec_buf, 10);
                    RS485_Direction(RS485_OUTPUT);
                    RS485_Puts((char *)rec_buf);
                    RS485_Direction(RS485_INPUT);
               }
          }
     }
}
```

8. How to use Jog Switch

8.1. Jog Switch API

Routine	Description
void Jog_Init(void)	- Initialize Jog Switch
Jog_t Jog_data	- Containing Jog state and Jog data
Jog_Rotate_t Jog_Get(Jog_t *data)	- Get Jog Switch of Rotation data
struct input_event key_code	- Check Jog Push data

8.1.1. void Jog_Init(void)

- Description

  This function is initialize Jog Switch.

- Header

  Jog.h

8.1.2. Jog_t Jog_data

- Description

  This structure value is check state of Jog.

- Struct Variable

  Jog_data.Jog_Count : Check on current Jog count value.

  Jog_data.Rotation : Zero-based of Jog count value, Increase or decrease the current state of Jog.

value	Meaning
Jog_Rotate_Increment	- Jog_Count > 0
Jog_Rotate_Decrement	- Jog_Count < 0
Jog_Rotate_Nothing	- Jog_Count == 0

  Jog_data.Absolute : Zero-based, Increase or decrease the current state of Jog

  Jog_data.Diff : Check on how many changing value to increase or decrease.

value	Meaning
+ value	- Jog rotate clockwise direction
- value	- Jog rotate counterclockwise direction

- Header

  Jog.h

8.1.3. Jog_Rotate_t Jog_Get(Jog_t *data)

- Description

  This value get rotation data of Jog switch.

- Parameter

  *data : Send to Jog data contacting address.

- Return Value

  Jog_Rotate_t : Return to current rotation state.

value	Meaning
Jog_Rotate_Increment	- Return 0 value
Jog_Rotate_Decrement	- Return 1 value
Jog_Rotate_Nothing	- Return 2 value

- Header

  Jog.h

8.1.4. struct input_event key_code

- Description

  This structure is check Jog push data.

- Struct Variable

  key_code.code : Define the KEY_SCROLLOCK value.

  key_code.value : Get Jog push state value.

value	Meaning
0	- Push Button Up
1	- Push Button Down

- Header

  Jog.h

8.2. Example

Jog Switch Example is composed of 2cases.

- Absolute : Check on increase or decrease state.
- Incremental : Check on Jog count value.

```
#include "stm32f4xx.h"
#include "Jog.h"
#include "Display.h"


int main(void)
{
```

```
    char put[100] = {0, };

    Jog_Init();
    Display_Init();

    while(1) {
        memset(put, 0, sizeof(put) );
#if 1// Absolute
        Jog_Get(&Jog_data);
        if ( Jog_data.Rotation == Jog_Rotate_Increment ) {
            if ( Jog_data.Diff != 0 ) {
                sprintf(put, "Jog Increment : %d, Diff : %d", Jog_data.Absolute, Jog_data.Diff);
                GUI_DispStringAtCEOL(put, 0, 0);
            }
        } else if ( Jog_data.Rotation == Jog_Rotate_Decrement ) {
            if ( Jog_data.Diff != 0 ) {
                sprintf(put, "Jog Decrement : %d, Diff : %d", Jog_data.Absolute, Jog_data.Diff);
                GUI_DispStringAtCEOL(put, 0, 0);
            }
        } else if ( Jog_data.Rotation == Jog_Rotate_Nothing) {
            if ( Jog_data.Diff != 0 ) {
                sprintf(put, "Jog Noting : %d, Diff : %d", Jog_data.Absolute, Jog_data.Diff);
                GUI_DispStringAtCEOL(put, 0, 0);
            }
        }

        // Jog Push
        if ( key_code.value == 1 ) {
            if ( key_code.code == KEY_SCROLLLOCK ) {
                Jog_data.Jog_Count = 0;
            }
        }
#else// Incremental
        // Jog Push
        if ( key_code.value == 1 ) {
            if ( key_code.code == KEY_SCROLLLOCK ) {
                Jog_data.Jog_Count = 0;
            }
        }
        sprintf(put, "Jog Data : %d", Jog_data.Jog_Count);
        GUI_DispStringAtCEOL(put, 0, 0);
#endif
    }
}
```

9. How to use the RTC

9.1. RTC API

Routine	Description
Daincube RTC API	
void EXTN_RTC_Init(void)	- Initialize and backup RTC
STM32 RTC API	
struct RTC_DateTypeDef RTC_Date	- Containing date information
struct RTC_TimeTypeDef RTC_Time	- Containing hour information
ErrorStatus RTC_SetDate(uint32_t RTC_Format, RTC_DateTypeDef *RTC_DateStruct)	- Error check setting date
ErrorStatus RTC_SetTime(uint32_t RTC_Format, RTC_TimeTypeDef *RTC_TimeStruct)	- Error check setting time
void RTC_GetDate(uint32_t RTC_Format, RTC_DateTypeDef *RTC_DateStruct)	- Get current date
void RTC_GetTime(uint32_t RTC_Format, RTC_TimeTypeDef *RTC_TimeStruct)	- Get current time

9.1.1. void EXTN_RTC_Init(void)

- Description

  This function is initialize external RTC clock. Success to initialize RTC then backup RTC value. If System power was ON/OFF, check initialize process. If initialize process is done, get backup RTC data.

- Header

  RTC.h

9.1.2. struct RTC_DateTypeDef RTC_Date

- Description

  RTC_DateTypeDef is containing year/month/date. This function locate in STM32 RTC API.

- Struct Variable

  RTC_Date.RTC_Year : Set year (0 ~ 99)

  RTC_Date.RTC_Month : Set month (1 ~ 12)

  RTC_Date.RTC_Date : Set date (1 ~ 31)

  RTC_Date.RTC_WeekDay : Set day of week

value	Meaning
RTC_Weekday_Monday	- (uint8_t)0x01

RTC_Weekday_Tuesday	- (uint8_t)0x02
RTC_Weekday_Wednesday	- (uint8_t)0x03
RTC_Weekday_Thursday	- (uint8_t)0x04
RTC_Weekday_Friday	- (uint8_t)0x05
RTC_Weekday_Saturday	- (uint8_t)0x06
RTC_Weekday_Sunday	- (uint8_t)0x07

- Header

  Stm32f4xx_rtc.h

9.1.3. struct RTC_TimeTypeDef RTC_Time

- Description

  RTC_DateTypeDef is containing hours/minutes/seconds. This function locate in STM32 RTC API.

- Struct Variable

  RTC_Time.RTC_Hours : Set hours (0 ~ 23)

  RTC_Time.RTC_Minutes : Set minutes (0 ~ 59)

  RTC_Time.RTC_Seconds : Set seconds (0 ~ 59)

- Header

  Stm32f4xx_rtc.h

9.1.4. ErrorStatus RTC_SetDate(uint32_t RTC_Format, RTC_DateTypeDef *RTC_DateStruct)

- Description

  This value is error check on setting date.

- Parameter

  RTC_Format : Set RTC Format. (DTP3 use RTC_Format_BIN)

  *RTC_DateStruct : Pointer to the structure value containing the date data.

- Return Value

  ErrorStatus : Return to error state..

value	Meaning
0	- ERROR
1	- SUCCESS

- Header

  Stm32f4xx_rtc.h

9.1.5. ErrorStatus RTC_SetTime(uint32_t RTC_Format, RTC_TimeTypeDef *RTC_TimeStruct)

- Description

  This value is error check on setting time.

- Parameter

  RTC_Format : Set RTC Format. (DTP3 use RTC_Format_BIN)

  *RTC_TimeStruct : Pointer to the structure value containing the time data.

- Return Value

  ErrorStatus : Return to error state.

value	Meaning
0	- ERROR
1	- SUCCESS

- Header

  Stm32f4xx_rtc.h

9.1.6. void RTC_GetDate(uint32_t RTC_Format, RTC_DateTypeDef *RTC_DateStruct)

- Description

  This function get current date.

- Parameter

  RTC_Format : Set RTC Format. (DTP3 use RTC_Format_BIN)

  *RTC_DateStruct : Pointer to the structure value containing the date data.

- Header

  Stm32f4xx_rtc.h

9.1.7. void RTC_GetTime(uint32_t RTC_Format, RTC_TimeTypeDef *RTC_TimeStruct)

- Description

  This function is get current date.

- Parameter

  RTC_Format : Set RTC Format. (DTP3 use RTC_Format_BIN)

  *RTC_TimeStruct : Pointer to the structure value containing the time data.

- Header

  Stm32f4xx_rtc.h

9.2. Example

```
#include "stm32f4xx.h"
#include "Display.h"
#include "KPD.h"
#include "RTC.h"


int main(void)
{
     RTC_DateTypeDef RTC_Date;
     RTC_TimeTypeDef RTC_Time;
     static int8_t date, month, year;
     static int8_t sec, min, hour;
     char put[100] = {0,};

     KPD_Init();
     Display_Init();
     EXTN_RTC_Init();

     while(1) {
          if ( key_code.value == 1 ) {
               if ( key_code.code == KEY_F1 ) {
                    RTC_Date.RTC_Date = 12;
                    RTC_Date.RTC_Month = 8;
                    RTC_Date.RTC_Year = 15;

                    RTC_Time.RTC_Seconds = 55;
                    RTC_Time.RTC_Minutes = 59;
                    RTC_Time.RTC_Hours = 23;

                    RTC_SetDate(RTC_Format_BIN, &RTC_Date);
                    RTC_SetTime(RTC_Format_BIN, &RTC_Time);
               }
          }
          RTC_GetDate(RTC_Format_BIN, &RTC_Date);
          RTC_GetTime(RTC_Format_BIN, &RTC_Time);

          date = RTC_Date.RTC_Date;
          month= RTC_Date.RTC_Month;
          year = RTC_Date.RTC_Year;

          sec  = RTC_Time.RTC_Seconds;
          min  = RTC_Time.RTC_Minutes;
          hour = RTC_Time.RTC_Hours;

          memset(put, 0, sizeof(put) );
          sprintf(put, "20%02d-%02d-%02d %02d:%02d:%02d", year, month, date, hour, min, sec);
          GUI_DispStringAtCEOL(put, 0, 0);
     }
}
```

10. How to use the USB MSC HOST

10.1. USB MSC HOST API

STM32F4 Discovery Lib containing USB MSC HOST API.

Routine	Description
TM_USB_MSCHOST_Result_t TM_USB_MSCHOST_Init(void)	- Initialize USB MSC HOST
void TM_USB_MSCHOST_Process(void)	- Operate USB HOST Core
TM_USB_MSCHOST_Result_t TM_USB_MSCHOST_Device(void)	- Check current USB device state
FRESULT f_mount (FATFS *fs, const TCHAR *path, BYTE opt)	- Mount/Unmount to USB
FRESULT f_open (FIL *fp, const TCHAR *path, BYTE mode)	- Open USB file
FRESULT f_close (FIL *fp)	- Close USB file
FRESULT TM_FATFS_USBDriveSize(uint32_t *total, uint32_t *free)	- Check USB storage
int f_putc (TCHAR c, FIL *fp)	- Write character USB
int f_puts (const TCHAR *str, FIL *cp)	- Write sting USB
TCHAR* f_gets (TCHAR *buff, int len, FIL *fp)	- Get length of data on USB

10.1.1. TM_USB_MSCHOST_Result_t TM_USB_MSCHOST_Init(void)

- Description

  This function is initialize USB MSC HOST.

- Return Value

  TM_USB_MSCHOST_Result_t : Return to USB MSC HOST state.

value	Meaning
TM_USB_MSCHOST_Result_Error	- An error occurred
TM_USB_MSCHOST_Result_Connected	- Device is connected and ready to use with FATFS
TM_USB_MSCHOST_Result_Disconnected	- Device is not connected
TM_USB_MSCHOST_Result_DeviceNotSupported	- Device is not supported
TM_USB_MSCHOST_Result_WriteProtected	- Device is write protected
TM_USB_MSCHOST_Result_LibraryNotInitialized	- Library is not initialized yet

- Header

  tm_stm32f4_usb_msc_host.h

10.1.2. void TM_USB_MSCHOST_Process(void)

- Description

  This function is operate USB HOST Core.

  TM_USB_MSCHOST_Process() function is periodically operate.

- Header

  tm_stm32f4_usb_msc_host.h

10.1.3. TM_USB_MSCHOST_Result_t TM_USB_MSCHOST_Device(void)

- Description

  This function is check on current USB device state.

- Return Value

  TM_USB_MSCHOST_Result_t : Return to USB MSC HOST state.

value	Meaning
TM_USB_MSCHOST_Result_Error	- An error occurred
TM_USB_MSCHOST_Result_Connected	- Device is connected and ready to use with FATFS
TM_USB_MSCHOST_Result_Disconnected	- Device is not connected
TM_USB_MSCHOST_Result_DeviceNotSupported	- Device is not supported
TM_USB_MSCHOST_Result_WriteProtected	- Device is write protected
TM_USB_MSCHOST_Result_LibraryNotInitialized	- Library is not initialized yet

- Header

  tm_stm32f4_usb_msc_host.h

10.1.4. FRESULT f_mount (FATFS *fs, const TCHAR *path, BYTE opt)

- Description

  This value is status of Mount/Unmount to USB

- Parameter

  *fs : Pointer to FATFS structure values containing USB file system.

  (If pointer in NULL, unmounts function operate.)

  *path : Define USB device number.

  opt : Set Mount/Unmount time.

value	Meaning
0	- Delay Mount/Unmount
1	- Immediately Mount/Unmount

- Return Value

FRESULT : Return to file function's value.

value	Meaning
FR_OK	- Succeeded
FR_DISK_ERR	- A hard error occurred in the low level disk I/O layer
FR_INT_ERR	- Assertion failed
FR_NOT_READY	- The physical drive cannot work
FR_NO_FILE	- Could not find the file
FR_NO_PATH	- Could not find the path
FR_INVALID_NAME	- The path name format is invalid
FR_DENIED	- Access denied due to prohibited access or directory full
FR_EXIST	- Access denied due to prohibited access
FR_INVALID_OBJECT	- The file/directory object is invalid
FR_WRITE_PROTECTED	- The physical drive is write protected
FR_INVALID_DRIVE	- The logical drive number is invalid
FR_NOT_ENABLED	- The volume has no work area
FR_NO_FILESYSTEM	- There is no valid FAT volume
FR_MKFS_ABORTED	- The f_mkfs() aborted due to any parameter error
FR_TIMEOUT	- Could not get a grant to access the volume within defined period
FR_LOCKED	- The operation is rejected according to the file sharing policy
FR_NOT_ENOUGH_CORE	- LFN working buffer could not be allocated
FR_TOO_MANY_OPEN_FILES	- Number of open files > _FS_SHARE
FR_INVALID_PARAMETER	- Given parameter is invalid

- Header

Ff.h

10.1.5.FRESULT f_open (FIL *fp, const TCHAR *path, BYTE mode)

- Description

This value is result of open USB file.

- Parameter

*fp : Pointer to FIL structure value containing open file information.

*path : Pointer to Open File name.

mode : File access mode and File open mode.

value	Meaning
FA_READ	- Access permission of File read.
FA_OPEN_EXISTING	- Open File(If not exist File, File open fail)
FA_WRITE	- Access permission of File write.
FA_CREATE_NEW	- Create File(If not exist File, File create fail)
FA_CREATE_ALWAYS	- Create File(If exist File, operate overwrite)
FA_OPEN_ALWAYS	- Open File(If not exist File, make File)

- Return Value

FRESULT : Return to file function's value.

value	Meaning
FR_OK	- Succeeded
FR_DISK_ERR	- A hard error occurred in the low level disk I/O layer
FR_INT_ERR	- Assertion failed
FR_NOT_READY	- The physical drive cannot work
FR_NO_FILE	- Could not find the file
FR_NO_PATH	- Could not find the path
FR_INVALID_NAME	- The path name format is invalid
FR_DENIED	- Access denied due to prohibited access or directory full
FR_EXIST	- Access denied due to prohibited access
FR_INVALID_OBJECT	- The file/directory object is invalid
FR_WRITE_PROTECTED	- The physical drive is write protected
FR_INVALID_DRIVE	- The logical drive number is invalid
FR_NOT_ENABLED	- The volume has no work area
FR_NO_FILESYSTEM	- There is no valid FAT volume
FR_MKFS_ABORTED	- The f_mkfs() aborted due to any parameter error
FR_TIMEOUT	- Could not get a grant to access the volume within defined period
FR_LOCKED	- The operation is rejected according to the file sharing policy
FR_NOT_ENOUGH_CORE	- LFN working buffer could not be allocated
FR_TOO_MANY_OPEN_FILES	- Number of open files > _FS_SHARE
FR_INVALID_PARAMETER	- Given parameter is invalid

- Header

Ff.h

10.1.6.FRESULT f_close (FIL *fp)

- Description

This value is result of close USB file.

- Parameter

*fp : Pointer to FIL structure value containing open file information.

- Return Value

FRESULT : Return to file function's value.

value	Meaning
FR_OK	- Succeeded
FR_DISK_ERR	- A hard error occurred in the low level disk I/O layer
FR_INT_ERR	- Assertion failed
FR_NOT_READY	- The physical drive cannot work
FR_NO_FILE	- Could not find the file
FR_NO_PATH	- Could not find the path
FR_INVALID_NAME	- The path name format is invalid
FR_DENIED	- Access denied due to prohibited access or directory full
FR_EXIST	- Access denied due to prohibited access
FR_INVALID_OBJECT	- The file/directory object is invalid
FR_WRITE_PROTECTED	- The physical drive is write protected
FR_INVALID_DRIVE	- The logical drive number is invalid
FR_NOT_ENABLED	- The volume has no work area
FR_NO_FILESYSTEM	- There is no valid FAT volume
FR_MKFS_ABORTED	- The f_mkfs() aborted due to any parameter error
FR_TIMEOUT	- Could not get a grant to access the volume within defined period
FR_LOCKED	- The operation is rejected according to the file sharing policy
FR_NOT_ENOUGH_CORE	- LFN working buffer could not be allocated
FR_TOO_MANY_OPEN_FILES	- Number of open files > _FS_SHARE
FR_INVALID_PARAMETER	- Given parameter is invalid

- Header

Ff.h

10.1.7.FRESULT TM_FATFS_USBDriveSize(uint32_t *total, uint32_t *free)

- Description

  This value is result of check USB storage.

- Parameter

  *total : Pointer to total storage.

  *free : Pointer to free storage.

- Return Value

  FRESULT : Return to file function's value.

value	Meaning
FR_OK	- Succeeded
FR_DISK_ERR	- A hard error occurred in the low level disk I/O layer
FR_INT_ERR	- Assertion failed
FR_NOT_READY	- The physical drive cannot work
FR_NO_FILE	- Could not find the file
FR_NO_PATH	- Could not find the path
FR_INVALID_NAME	- The path name format is invalid
FR_DENIED	- Access denied due to prohibited access or directory full
FR_EXIST	- Access denied due to prohibited access
FR_INVALID_OBJECT	- The file/directory object is invalid
FR_WRITE_PROTECTED	- The physical drive is write protected
FR_INVALID_DRIVE	- The logical drive number is invalid
FR_NOT_ENABLED	- The volume has no work area
FR_NO_FILESYSTEM	- There is no valid FAT volume
FR_MKFS_ABORTED	- The f_mkfs() aborted due to any parameter error
FR_TIMEOUT	- Could not get a grant to access the volume within defined period
FR_LOCKED	- The operation is rejected according to the file sharing policy
FR_NOT_ENOUGH_CORE	- LFN working buffer could not be allocated
FR_TOO_MANY_OPEN_FILES	- Number of open files > _FS_SHARE
FR_INVALID_PARAMETER	- Given parameter is invalid

- Header

  tm_stm32f4_fatfs.h

10.1.8. int f_putc (TCHAR c, FIL *fp)

- Description

  This value is result of write character USB.

- Parameter

  c : Pointer to write character for USB device.

  *fp : Pointer to FIL structure value containing open file information.

- Return Value

  int : Return to result of writing character.

value	Meaning
EOF	- Fail
N	- Length of character

- Header

  Ef.h

10.1.9. int f_puts (const TCHAR *str, FIL *cp)

- Description

  This value is result of write sting USB.

- Parameter

  *str : Pointer to write string for USB device.

  *fp : Pointer to FIL structure value containing open file information.

- Return Value

  int : Return to result of writing string.

value	Meaning
EOF	- Fail
N	- Length of string

- Header

  Ef.h

10.1.10. TCHAR* f_gets (TCHAR *buff, int len, FIL *fp)

- Description

  This value is get length of data on USB.

- Parameter

  *buff : Pointer to saving data value.

  len : Define length of data.

  *fp : Pointer to FIL structure value containing open file information.

- Return Value

  TCHAR* :

value	Meaning
EOF	- Fail
N	- Length of string

- Header

  Ef.h

10.2. Example

```
#include "stm32f4xx.h"
#include "ETC.h"
#include "tm_stm32f4_fatfs.h"
#include "tm_stm32f4_usb_msc_host.h"


int main(void)
{
    FATFS USB_Fs;
    FIL USB_Fil;
    char buffer[50];
    uint8_t write = 1;
    uint32_t free, total;

    ETC_Init();

    /* Initialize USB MSC HOST */
    TM_USB_MSCHOST_Init();

    while(1) {
        /* Host Task handler */
        /* This have to be called periodically as fast as possible */
        TM_USB_MSCHOST_Process();
        /* Device is connected and ready to use */
        if (TM_USB_MSCHOST_Device() == TM_USB_MSCHOST_Result_Connected) {
            /* If we didn't write data already */
            if (write) {
                /* Try to mount USB device */
                /* USB is at 1: */
                if (f_mount(&USB_Fs, "1:", 1) == FR_OK) {
                    LED_ON_OFF(LED_OFF, LED_ALARM);
                    LED_ON_OFF(LED_ON, LED_RUN);
                    /* Mounted ok */
                    /* Try to open USB file */
                    if (f_open(&USB_Fil, "1:DTP3_USB_TEST.txt", FA_READ | FA_WRITE | FA_OPEN_ALWAYS) == FR_OK) {
                        /* We want to write only once */
                        write = 0;

                        /* Get total and free space on USB */
                        TM_FATFS_USBDriveSize(&total, &free);

                        /* Put data */
                        f_puts("This is my first file with USB and FatFS₩n", &USB_Fil);
                        f_puts("with USB MSC HOST library from stm32f4-discovery.com₩n", &USB_Fil);
                        f_puts("--------------------------------------------------₩n", &USB_Fil);
                        f_puts("USB total and free space:₩n₩n", &USB_Fil);
                        /* Total space */
                        sprintf(buffer, "Total: %8u kB; %5u MB; %2u GB₩n", total, total / 1024, total / 1048576);
                        f_puts(buffer, &USB_Fil);
```

```
                            /* Free space */
                            sprintf(buffer, "Free:  %8u kB; %5u MB; %2u GB₩n", free, free / 1024, free / 1048576);
                            f_puts(buffer, &USB_Fil);
                            f_puts("-----------------------------------------------------₩n", &USB_Fil);
                            /* Close USB file */
                            f_close(&USB_Fil);

                            /* Turn GREEN LED On and RED LED Off */
                            /* Indicate successful write */
                            LED_ON_OFF(LED_OFF, LED_RUN);
                        }
                    }
                    /* Unmount USB */
                    f_mount(0, "1:", 1);
                }
            } else {
                /* Not inserted, turn on RED led */
                LED_ON_OFF(LED_ON, LED_ALARM);

                /* Ready to write next time */
                write = 1;
            }
        }
}
```

11. How to use the Font

11.1. How to use Font

DTP3 support various fonts. STemWin library contain various fonts. Fonts used 8bit ASCII ISO 8859 font.

If you apply fonts, you will refer to "Chapter 10. Fonts" in "STemWin5.pdf".

11.2. How to use External Font

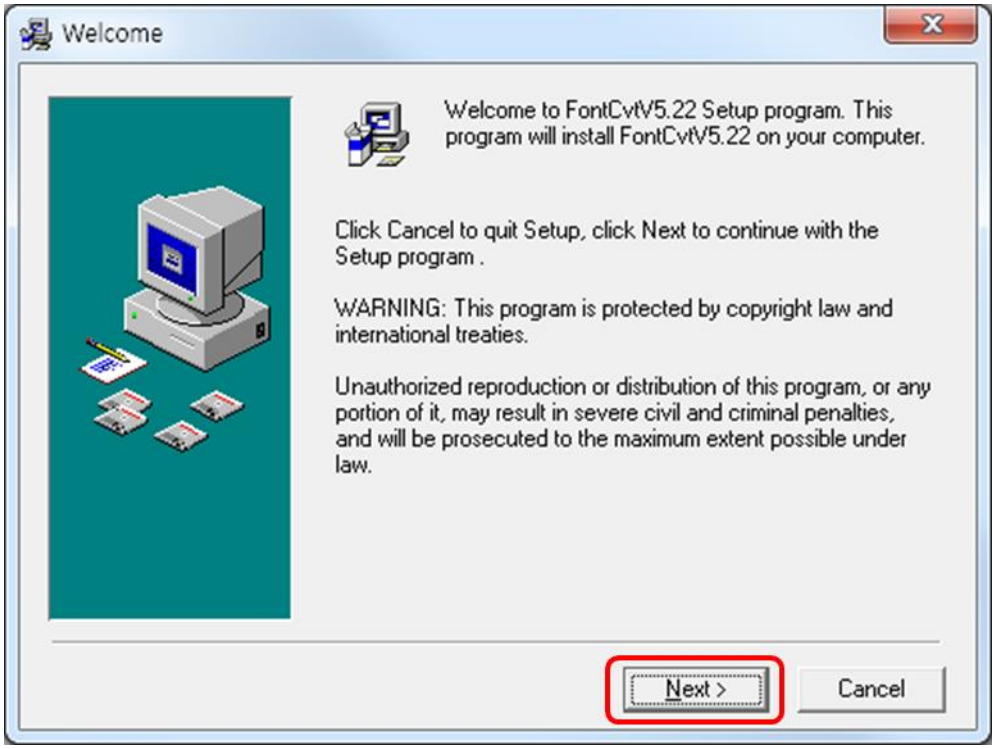
DTP3 support external fonts through STemWin library. If you apply external fonts, you execute "FontCvt V5.22" program.

And you will install "SetupFontCvt_V522.exe". "SetupFontCvt_V522.exe" is located "DTP3_DEVKITW02_DTP3_SWW03_SoftwareWSTemWin"

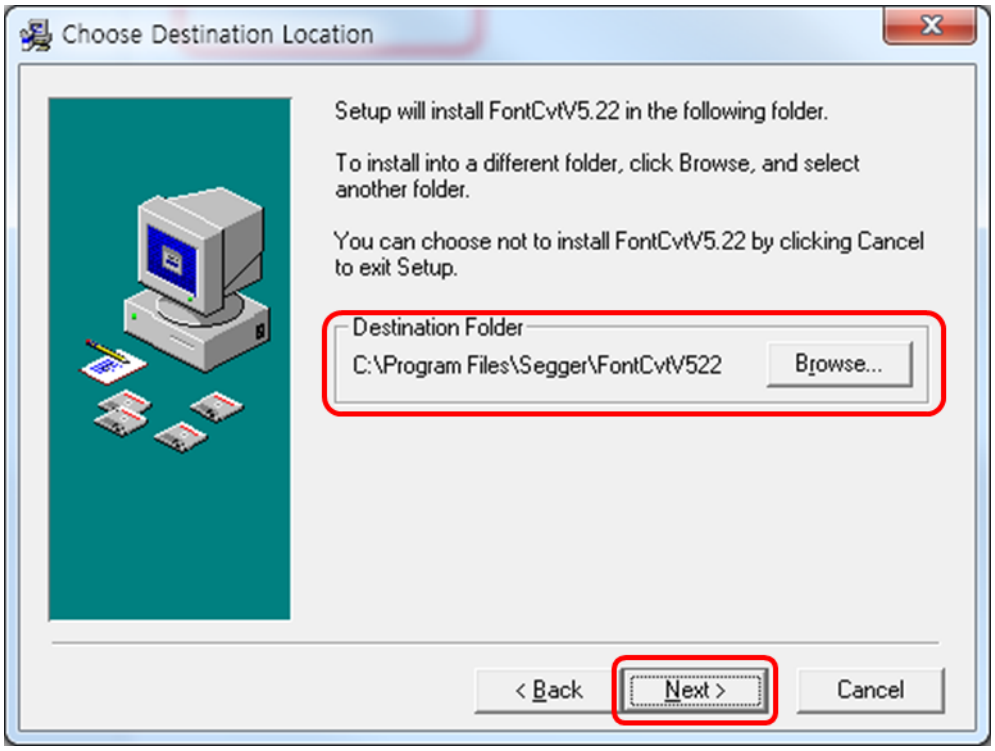
11.2.1. SetupFontCvt_V522.exe Install

Execute "SetupFontCvt_V522.exe."

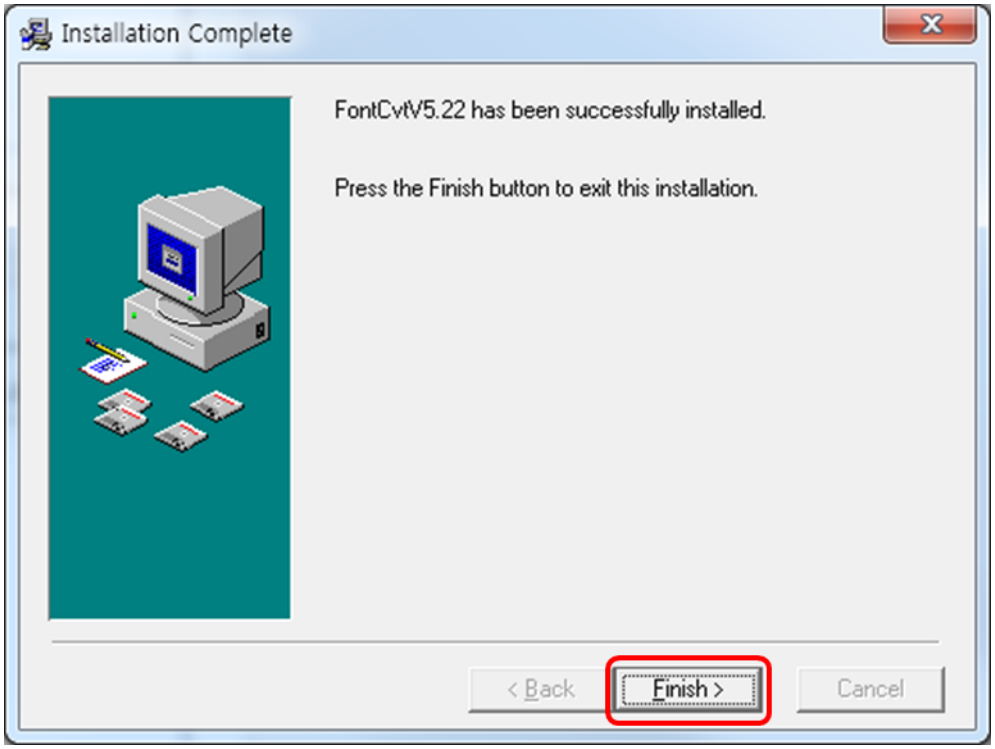
Click "Next" button.



Click "Next" button.



Program install finish, Click "finish" button.



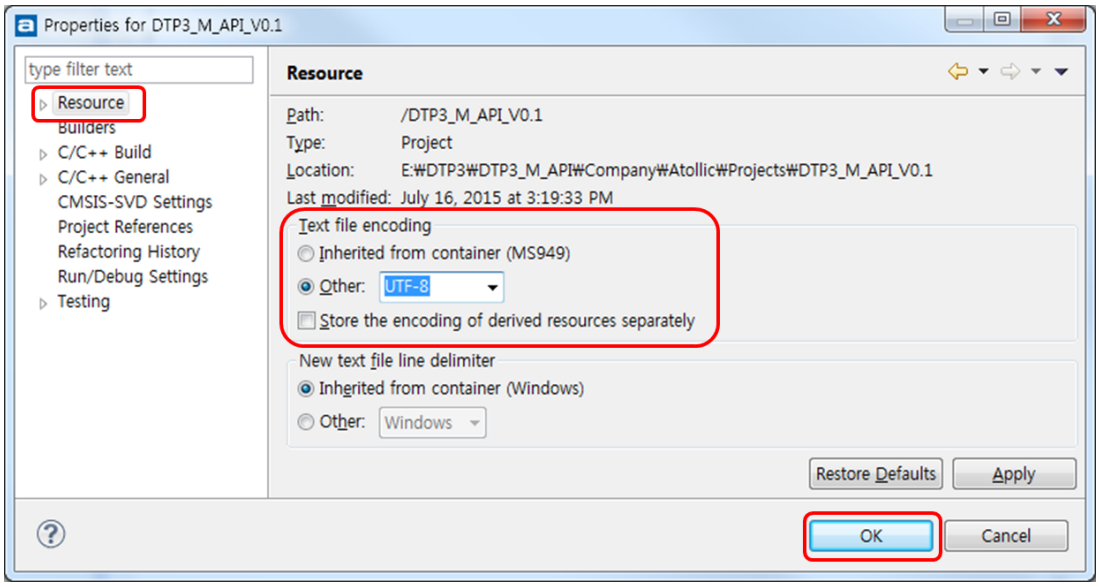
11.2.2.External Font Create

"FontCvt V5.22" program refer to "Chapter 11. Font Converter" in "STemWin5.pdf".

If you make 8bit ASCII ISO 8859 font, refer to Font API(p205).

If you make 16bit Unicode font, refer to "Chapter 29. Language Support" in "STemWin5.pdf" and uses Unicode API(p958).

If you use 16bit Unicode, Change project properties. Reference to below screenshot.



11.2.3. Example

```
#include "stm32f4xx.h"
#include "Display.h"
#include "font_test_16.h"


int main(void)
{
    Display_Init();

    GUI_SetFont(&GUI_Fontfont_test_16);
    GUI_UC_SetEncodeUTF8();

    GUI_DispString("(주)다인큐브");
    GUI_DispNextLine();
    GUI_DispString("公司作为一个立方体");
    GUI_DispNextLine();
    GUI_DispString("公司作为一个立方体");

    while(1) {

    }
}
```

12. FAQ.