M3H Group(2)
Application Note
Asynchronous Serial Communication Circuit
(UART-C)

Outlines
This application note is a reference material for developing products using the asynchronous serial communication circuit (UART) function of M3H Group (2).
This document helps the user check operation of the product and develop its program.

Target sample program: UART_Echo
# Table of Contents

Outlines ................................................................................................................................................. 1  
Table of Contents .................................................................................................................................. 2  
1. Preface .............................................................................................................................................. 3  
2. Reference Document ........................................................................................................................ 4  
3. Function to Use .................................................................................................................................. 4  
4. Target Device .................................................................................................................................... 4  
5. Operation confirmation condition ...................................................................................................... 5  
6. Evaluation Board Setting .................................................................................................................. 6  
7. Operation of Evaluation Board ......................................................................................................... 6  
8. Outline of UART interface function .................................................................................................. 7  
   8.1. Clock Supply ..................................................................................................................................... 7  
9. Sample Program ............................................................................................................................... 8  
   9.1. Initialization ....................................................................................................................................... 8  
   9.2. Sample Program Main Operation .................................................................................................. 8  
   9.3. Output Example of Sample Program .............................................................................................. 9  
      9.3.1. Setting Example of Terminal Software ...................................................................................... 10  
   9.4. Operating Flow of Sample program ............................................................................................. 11  
10. Precaution ..................................................................................................................................... 23  
11. Revision History ............................................................................................................................ 23  
RESTRICTIONS ON PRODUCT USE ............................................................................................... 24
1. Preface

This sample program is used to check the operation of UART communication function. It executes command input and echo back from the terminal software on the PC via the USB - UART interface.

Structure diagram of Sample program

![Structure diagram of Sample program]
2. Reference Document

- Datasheet
  TMPM3H group (2) datasheet Rev2.0 (Japanese edition)
- Reference manual
  Asynchronous serial communication circuit (UART-C) Rev2.0 (Japanese edition)
- Other reference document
  TMPM3H(2) Group Peripheral Driver User Manual (Doxygen)

3. Function to Use

<table>
<thead>
<tr>
<th>IP</th>
<th>channel</th>
<th>port</th>
<th>Function / operation mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asynchronous</td>
<td>ch0</td>
<td>PA1 (UT0TXDA)</td>
<td>UART mode</td>
</tr>
<tr>
<td>communication</td>
<td></td>
<td>PA2 (UT0RXD)</td>
<td></td>
</tr>
</tbody>
</table>

4. Target Device

The target devices of application note are as follows.

<table>
<thead>
<tr>
<th>TMPM3HQFDFG</th>
<th>TMPM3HQFZFG</th>
<th>TMPM3HQFYFG</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMPM3HPFDFG</td>
<td>TMPM3HPFZFG</td>
<td>TMPM3HPFYFG</td>
</tr>
<tr>
<td>TMPM3HNFDFG</td>
<td>TMPM3HNFZFG</td>
<td>TMPM3HNFYFG</td>
</tr>
<tr>
<td>TMPM3HNFDDFG</td>
<td>TMPM3HNFZDFG</td>
<td>TMPM3HNFYDFG</td>
</tr>
<tr>
<td>TMPM3HMFDFG</td>
<td>TMPM3HMFZFG</td>
<td>TMPM3HMFYFG</td>
</tr>
</tbody>
</table>

* This sample program operates on the evaluation board of TMPM3HQFDFG. If other function than the TMPM3HQ one is checked, it is necessary that CMSIS Core related files (C startup file and I/O header file) should be changed properly.

The BSP related file is dedicated to the evaluation board (TMPM3HQ). If other function than the TMPM3HQ one is checked, the BSP related file should be changed properly.
5. Operation confirmation condition

Used microcontroller         TMPM3HQFDFG
Used board                 TMPM3HQFDFG Evaluation Board (Product of Sensyst)
Unified development environment         IAR Embedded Workbench for ARM 8.11.2.13606
Unified development environment         μVision MDK Version 5.24.2.0
Terminal software           Tera Term V4.96
Sample program            V1100

For purchasing the board, refer to the following homepage. (http://www.chip1stop.com/)
6. Evaluation Board Setting

The following pin connections should be done on the evaluation board.

<table>
<thead>
<tr>
<th>CN12</th>
<th>Board function</th>
<th>Through hole No.</th>
<th>Through hole No.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>USB UART conversion</td>
<td>53 : PA1</td>
<td>54 : USB TXD</td>
</tr>
<tr>
<td></td>
<td>USB UART conversion</td>
<td>55 : PA2</td>
<td>56 : USB RXD</td>
</tr>
</tbody>
</table>

7. Operation of Evaluation Board

PC should be connected to the USB_UART terminal on the evaluation board with a USB cable. After start-up the terminal software (Tera Term), performs communication setting on the terminal software. Push the reset button on the evaluation board.

The following operations should be done on the terminal software (Tera Term) on PC.

1. Initial display
   "Input =" requires a user inputting proper information on the display
2. Character input
   The user should input proper characters. Then, "Enter" should be done.
3. Echo-back output
   The input characters are displayed after "Echo =".
4. When the count of the characters exceeds the maximum value;
   "Input Error !!" is displayed.
8. Outline of UART interface function

The asynchronous serial communication circuit (UART) can operate as a transmission / reception circuit of 1 channel (UTxTXDA / UTxTXDB / UTxRXD) per unit.

<table>
<thead>
<tr>
<th>Function classification</th>
<th>Function</th>
<th>Operation explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud rate control</td>
<td>Frequency dividing of prescaler</td>
<td>Selectable from 1/1 to 1/512 of the ΦT0 frequency.</td>
</tr>
<tr>
<td></td>
<td>Baud rate generator</td>
<td>N dividing or N + (64 - K) / 64 (N = 1 to 65535 and K = 0 to 63) dividing of the source clock frequency are possible.</td>
</tr>
<tr>
<td>Data format</td>
<td>Data length</td>
<td>Selectable 7, 8, or 9-bit.</td>
</tr>
<tr>
<td></td>
<td>Parity</td>
<td>Parity control: Enable or disable selection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Parity type: Even or odd parity is selectable</td>
</tr>
<tr>
<td></td>
<td>Stop bit length</td>
<td>Selectable 1-bit or 2-bit.</td>
</tr>
<tr>
<td></td>
<td>Data transfer order</td>
<td>Selectable LSB first or MSB first.</td>
</tr>
<tr>
<td></td>
<td>Data signal inversion</td>
<td>Inversion control of the input and output data signal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Selectable disabled or enabled.</td>
</tr>
<tr>
<td>Transmission</td>
<td>FIFO storage stages</td>
<td>Reception: 8 stages</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transmission: 8 stages</td>
</tr>
<tr>
<td></td>
<td>Noise cancelling function</td>
<td>Reception: Noise cancelling function is enabled or disabled for UTxRXD.</td>
</tr>
<tr>
<td></td>
<td>Error detection</td>
<td>Reception: Parity error, Framing error, Break error, Overrun error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transmission: Break error, Trigger transmission error</td>
</tr>
<tr>
<td></td>
<td>Handshake function</td>
<td>Transmission / reception control by handshake with UTxCTS_N / UTxRTS_N signal is possible.</td>
</tr>
<tr>
<td></td>
<td>Wake-up function</td>
<td>Serial link operation using the wakeup function in the 9-bit mode is possible.</td>
</tr>
<tr>
<td>Interlocking control</td>
<td>Interrupt</td>
<td>Reception interrupt, Transmission interrupt, and Error interrupt.</td>
</tr>
<tr>
<td></td>
<td>DMA request</td>
<td>Reception DMA request: Burst transfer or Single transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transmission DMA request: Burst transfer or Single transfer</td>
</tr>
<tr>
<td>Special control</td>
<td>Half clock mode (Transmission /</td>
<td>Transmission and reception with half width of &quot;0&quot; width of the normal UART waveform is possible.</td>
</tr>
<tr>
<td></td>
<td>reception)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Loopback function (Test function)</td>
<td>The transmission data is connected to the reception data and the loopback test is possible.</td>
</tr>
<tr>
<td></td>
<td>Software reset</td>
<td>Initialization can be done by software.</td>
</tr>
</tbody>
</table>

UTxTXDA can be exchanged for UTxRXD and UTxCTS_N can be exchanged for UTxRTS_N, too. This is done by a port setting. Refer to "Input/Output Ports" of the reference manual.

8.1. Clock Supply

When you use UART, please set an applicable clock enable bit to "1" (clock supply) in fsys supply stop register A ([CGFSYSENA]), fsys supply stop register B ([CGFSYSENB]), and fc supply stop registers ([CGFCEN]).

Please refer to "Clock Control and Operation Mode" of the reference manual for the details.

When attempting to stop supplying the clock, make sure to check whether the UART is stopping. Note that when the MCU enters STOP mode, make sure to check whether the UART is stopping as well.
9. Sample Program

The data transferred from the terminal software is echoed back.

9.1. Initialization

The following initialization is done after power is supplied. The port setting is executed after the initialization of each clock setting, the watchdog timer setting and the clock setting.

9.2. Sample Program Main Operation

After the initialization operation, shift to the main function and perform the following initialization.
1. Initialization of BSP (Board Support Package)
2. Initialization of variables
3. Initialize the driver
4. Initialize the application

After performing the above processing, start the timer and clear the data. It waits for input at "Input =" indication on the PC terminal software. After inputting, the input character is displayed following "Echo =" display as the echo back output.
9.3. Output Example of Sample Program

When the sample program operates, the command results are shown as follows:

```
Input = 1234567890
Echo = 1234567890
Input = 123456789012345678901234567890
Echo = 123456789012345678901234567890
Input = 123456789012345678901234567890123
Input Error !!
Input =
```
9.3.1. Setting Example of Terminal Software

The operation of the terminal software (Tera Term) has been checked with the following settings.
9.4. Operating Flow of Sample program

The operating flows of the sample program are shown in the following:
Creation and Initialization

- Ref BSP initialization
- RAM initialization
- Ref Driver initialization
- Ref Application initialization

Driver initialization

- Ref T32A Driver initialization
Although initialization setting of 32-bit timer event counter of TMPM3H is done, it is not used in actual operation of this sample program.

**Application initialization**

- **Timer Application initialization**
- **UART Application initialization**

**T32A Driver initialization**

- **BSP (Application)**
  - `bsp_get_timer_ch(BSP Timer)`
  - Channel number = `bsp_get_timer_ch(-)`
  - `t32a_mode_init(Instance address)`
  - `timer_initialize(Timer Instance address)`
  - `result = timer_initialize(-)`
  - `t32a_timer_init(Instance address)`
  - `result = t32a_timer_init(-)`
  - `result = t32a_mode_init(-):success`

- **T32A (Driver)**
  - Acquisition of Timer channel for 1-ms timer
UART Application initialization

- **BSP** (Application)
  - `bsp_get_usb_uart_tx_ch`
  - **Acquisition of Channel for USB UART**

**UART IO Initial setting creation**

- `uart_io_initialize` (Storage destination of UART IO initial setting)
  - Assignment of Address to Specified UART CH register
  - **Create UART Driver initial setting**
    - `uart_initial_setting_t`
  - **Registration of UART driver event handler**
    - Transmission event handler
    - Reception event handler
  - `uart_init(UART instance address)`
  - `uart_receiveIt(UART instance address, Storage destination of Reception buffer information)`

- **Exception enable**
  - `INTUARTxTX`
  - `INTUARTxRX`
  - `INTUARTxERR`

**UART IO instance address**

- UART IO instance address = `uart_io_initialize()`
T32A Driver end

- Release of assignment of Register address to Channel for 1-ms timer

Application end

- ref UART Application end
- ref Timer Application end
uart_io_finalize(UART IO instance address)

uart_deinit(UART IO instance address)

Exception disable:
- INTUARTxTX
- INTUARTxRX
- INTUARTxERR

uart_io_finalize(-)

Release of assignment of Address to Specified UART CH register
Start-up

Stop

ref Timer Application start-up

ref Timer Application stop
“printf” is retargeted to “fputc”, and the characters are output one by one.
"getchar" is retargeted to "getc", and the characters are input one by one.

Pick up Reception data.

[Reception result! = Failure]

[Reception information = Reception data is present.]
Transmission

irq_usb_uart_tx

uart_io_tx_irq_handler (UART IO instance address)

uart_transmit_irq_handler (UART instance address)

Registration transmission handler (UART instance address, Transmission result)

Transmission result is stored.

Transmission result is stored.

Registration transmission handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_irq_handler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)

uart_io_tx_irq_handler(-)

uart_transmit_IRQHandler(-)
Reception

uart_io_rx_irq_handler (UART IO instance address)

uart_receive_irq_handler (UART instance address)

Registration reception handler
(UART instance handle, Reception result,
Storage destination of Reception information)

Reception result and
Reception information are stored.

Registration transmission handler(-)

uart_receive_irq_handler(-)

uart_io_rx_irq_handler(-)

uart_rx_irq_handler (UART IO instance address)
10. Precaution
When using the sample program with CPU other than TMPM3HQ, please check operation sufficiently.

11. Revision History

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2018-05-22</td>
<td>-</td>
<td>First release</td>
</tr>
</tbody>
</table>

RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as “TOSHIBA”. Hardware, software and systems described in this document are collectively referred to as “Product”.

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.

- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA’s written permission, reproduction is permissible only if reproduction is without alteration/omission.

- Though TOSHIBA works continually to improve Product’s quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the “TOSHIBA Semiconductor Reliability Handbook” and (b) the instructions for the application with which the Product will be used with or for.

Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS’ PRODUCT DESIGN OR APPLICATIONS.

- PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT (“UNINTENDED USE”). Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, devices related to electric power, and equipment used in finance-related fields. IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT. For details, please contact your TOSHIBA sales representative.

- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.

- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.

- ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.

- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.

- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.