Operating the MU-4 in transparent mode

(Using UART to transmit / receive data)

Dec. 17, 2021
Circuit Design, Inc.
Ver. 1.0

Introduction

This application note explains in detail how to transmit and receive data in UART format while in MU-4 transparent mode. Using asynchronous UART has the advantage of being able to easily detect the data starting position which is an important feature in radio communication.

Please refer to the MU-4 operation guide for information on setting the transparent mode and details on command timing.

How to generate data using UART for radio communication

For transmitting and receiving serial data while the MU-4 is in transparent mode, it is important to create data in a manner that can be transmitted reliably. Detailed procedures and tips regarding this are written below.

1. Channel information
   The frequency channel set in the receiver will also receive any data present in any nearby frequency channels. If allowed to receive data from these channels, the reception quality will fall and data reception will become unstable.
   To avoid this, the transmitter channel information is included in the data so that any data not from this channel can be ignored by the receiver.

2. Inserting a delay when switching from transmission to reception
   In the case of MU-4, it takes enough time to issue the @TRRX command, so it is not necessary to set any dummy byte.

   However in general, when using the CPU's UART and preparing the transmission data, in actuality it takes about 2.1ms at 4800 bps from writing 1 byte of data in the UART buffer to the completion of transmission to the TXD terminal. In this case, when switching to receive immediately after the last data has been written to the UART buffer during transmit - there is the possibility that this data does not get fully transmitted. To avoid this, before switching to reception a delay can be inserted after writing the last data in the UART buffer. This can be achieved by writing a 2 byte dummy data following the last data.

3. Sending preamble data
   In order to stabilise the DC to determine logic level and set the data receiving clock correctly etc., please insert a preamble sequence, 11001100 of duration 15ms before the first transmit data. In the case where the data rate is 4800 bps, send 9 bytes containing 0xCC as the preamble sequence.
   In the case of MU-4, if the preamble is sent after receiving the response for @TRTX, there will be nearly 10ms HIGH level before the preamble is transmitted. This can cause the 4ms HIGH interrupt to occur at the receiver side. Therefore please start transmitting the preamble immediately after the @TRTX command is issued without verifying the response for the command.

4. Sending DC HIGH level
   After sending the 15ms preamble sequence (in the case when 4800 bps is the RF data rate, send 9 bytes containing 0xCC), then transmit a HIGH level of duration 5ms before the first transmission data.
   During the 5ms HIGH level there is no UART data which easily allows the transmitted data start point to be detected.

   After transmitting the preamble data, disable the UART transmit interrupt and enable it again after the transmission of the 5ms HIGH level. (for the detailed procedure, refer to the example flowchart showing transmission of data using UART)
   At the receiver side, during the reception of 4ms HIGH level, set the UART interrupt to disable after which set the UART interrupt to enable to allow reception at the start of the data.
   Details regarding the method for detecting the data start position will be described later.

5. Identification code
   Radio communication is not confined to a fixed receiving station as it is in wired communication. To allow the receiving station to confirm the intended reception, it is necessary to insert identification code in the transmission data.

6. Error detection code
   During radio communication, various factors can greatly affect reception level causing data to be corrupted. To ensure detection of corrupted data during receive, a checksum or CRC error detection code is calculated on each byte at the transmitter and the result is appended at the end of the transmitted data. This allows the receiver to perform error detection upon reception. In some cases, error correction codes can be incorporated to improve reliability.

   Example formula to generate CRC code:
   X^16 +X^13 +X^12 +X^11 +X^10 +X^8 +X^6 +X^5 +X^2 + 1
   Polynomial = 0x3D65
   For a data length of 10 bytes (that includes a 2 byte CRC), Hamming distance of 6 can be achieved.
   (Errors of up to 5 bits can be detected)

7. Data length
   If the transmitted data length varies, then include this in the transmitted data to indicate its total length. If the data length is fixed, then this is not necessary.

Refer to the transmission data format example below (in the case of variable data length)
UART settings: Start bit = 1, Data length = 8 bits, Stop bit = 1, no parity, data rate = 4800 bps

③ Preamble (11001100..) 15 ms	④ "H" level 5ms	① Freq. Ch 1 byte	⑤ ID code 1 1 byte	⑤ ID code 2 1 byte	⑤ ID code 3 1 byte

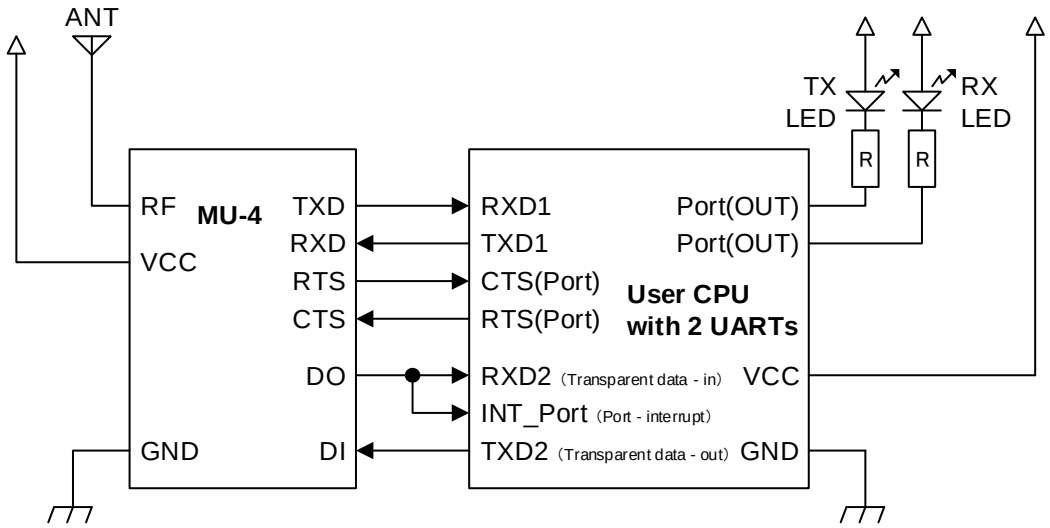
⑤ ID code 4 1 byte	⑦ Data length 1 byte	Data 1 1 byte	Data 2 1 byte	…	Data N 1 byte	⑥ CRC High byte	⑥ CRC Lower byte

* If the transmitted data is cyclic (repeating), the 5ms HIGH level is set after transmission of the CRC Lower byte for all subsequent data.

Detecting the start of data

By using asynchronous UART for radio transmission makes it easy to detect the start position of the data. For a detailed procedure, refer to the example flow chart at the end of this document along with the following explanation.

To begin with, a diagram showing an example of how the CPU is connected to the MU-4 is shown below. Please make sure that the output corresponding to the transparent mode on the MU-4 is connected to both the CPU's UART RXD and the INT_PORT (configured as an external input interrupt).



Example connection of MU-4 with CPU when using transparent mode

Data reception

The procedure to receive data is explained below.

1. Set UART receive interrupt to disabled and set the external input interrupt for the rising and falling edge to enabled.

2. If the interrupt occurs due to a rising or falling edge and the INT_Port is HIGH, the 4ms timer is initialised and counting starts.

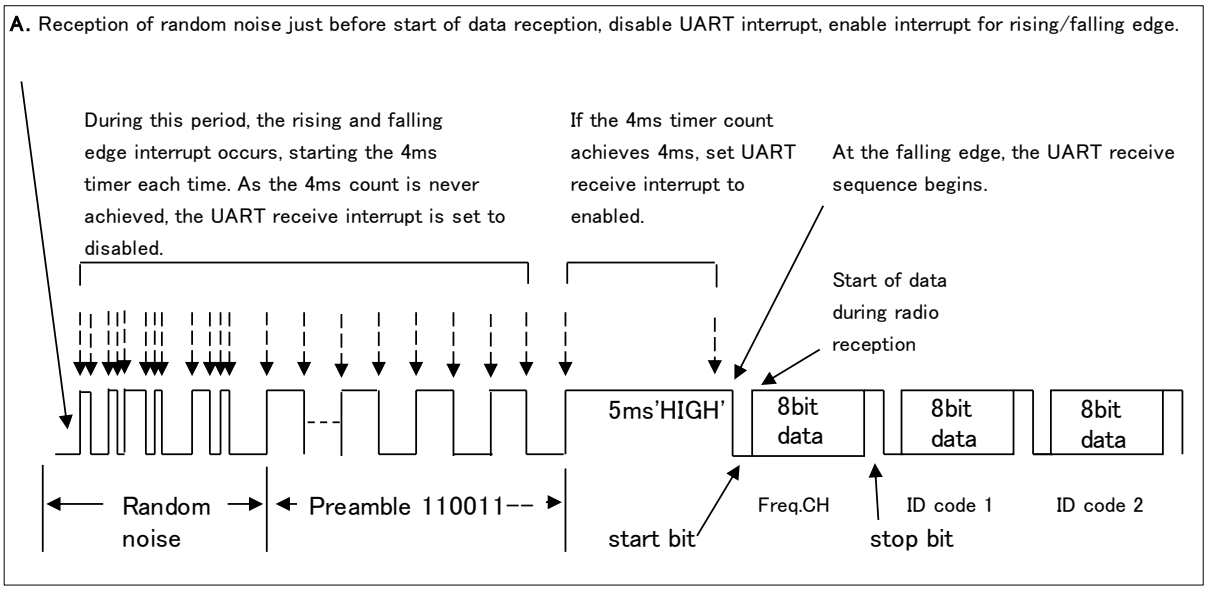
3. While the 4ms timer is running, if the interrupt occurs due to a rising or falling edge the 4ms timer is stopped and re-initialised. Then the process is repeated from step 2.

4. If the timer is allowed to reach 4ms with no rising or falling edge and no interrupt occurs, set the interrupt for the rising/falling edge to disabled and set UART receive interrupt to enabled.

5. After the enabling of the UART receive interrupt, a subsequent falling edge indicates the start bit for the UART data. After the presence of 8 further bits and the HIGH level STOP bit, the UART receive interrupt occurs indicating reception of the first data.

6. As data reception continues, the UART receive interrupt occurs repeatedly until reception of the error detection data is completed. Following this set the UART receive interrupt to disabled and the external input interrupt to enabled and repeat the process from step 2.

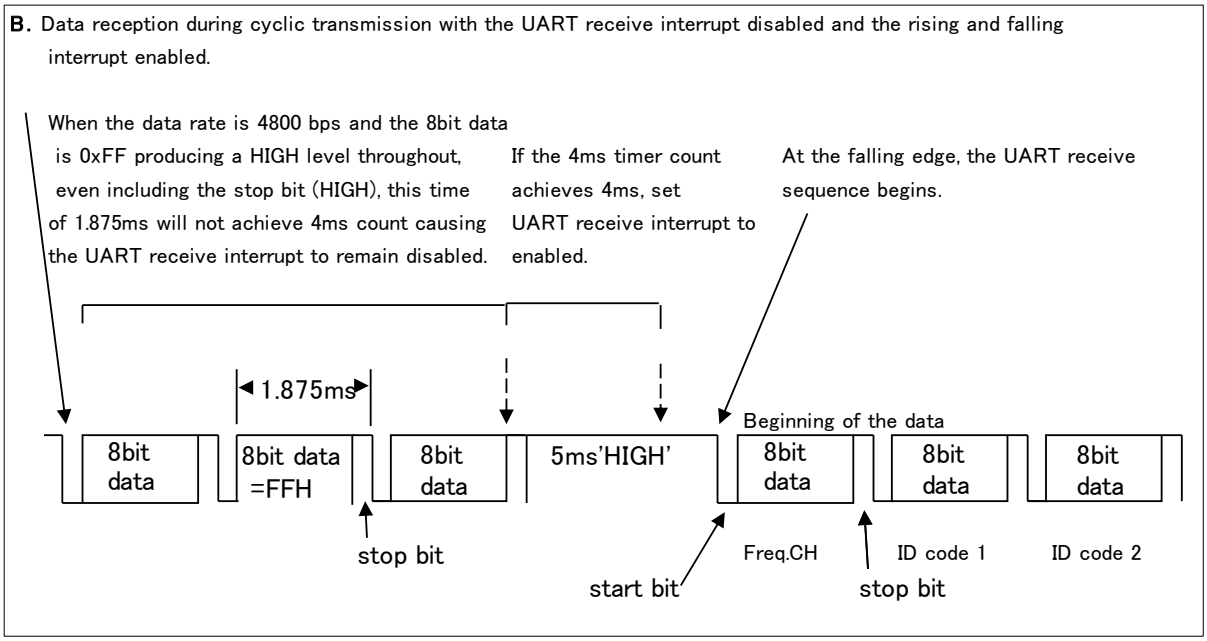
7. In the event that the UART receive error interrupt occurs, set the UART receive interrupt to disabled and the external input interrupt to enabled again. Then repeat process from step 2.



Going from random noise to start of data reception

When a HIGH level lasting more than 4ms is detected inside the presence of random noise, set the UART receive interrupt to enabled so the UART data reception begins at the next falling edge. Consequently, as the UART receive error interrupt occurs, set the UART receive interrupt to disabled and the external input interrupt to enabled. Then the process repeats from step 2.

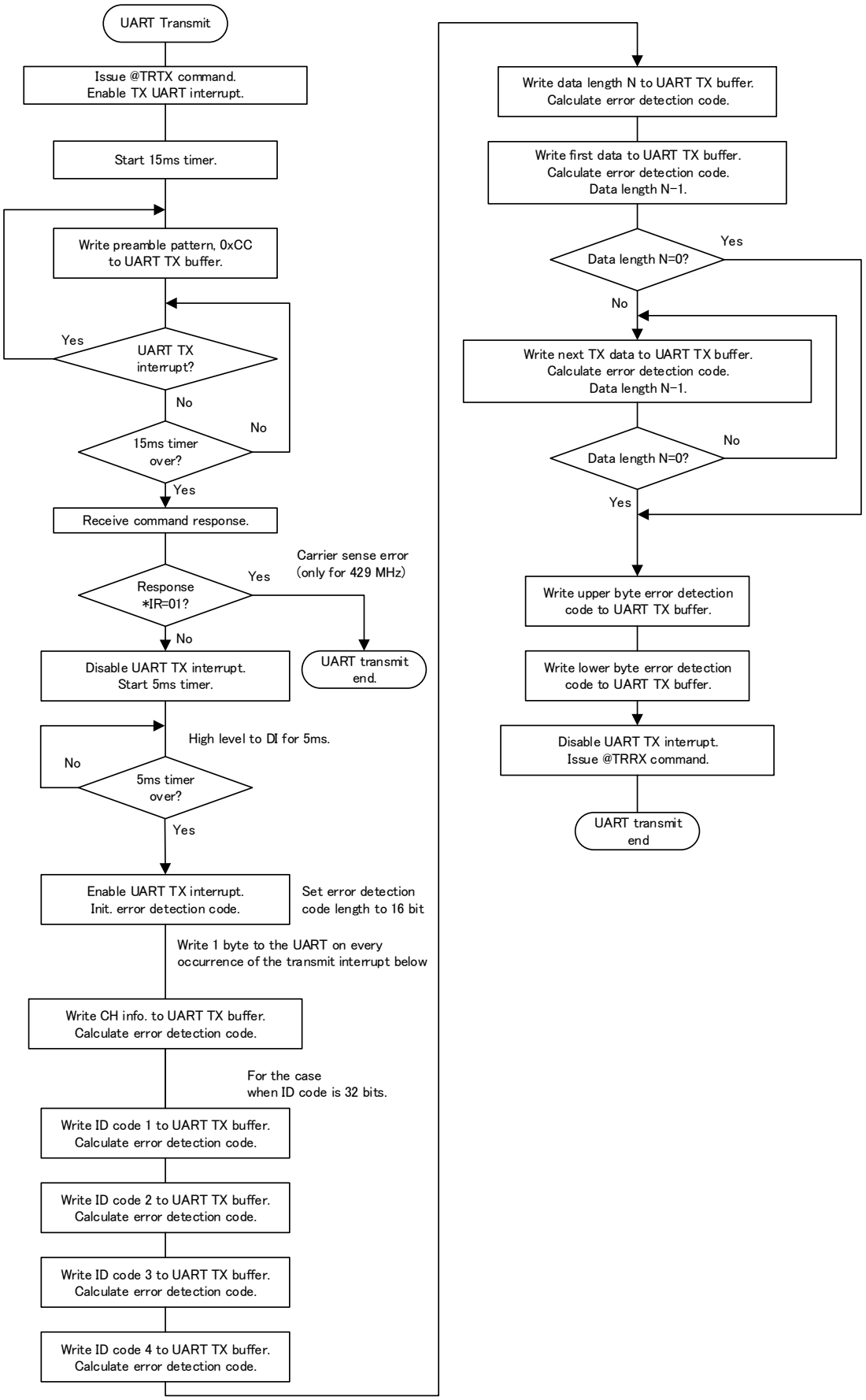
In the case that the transmitted data is cyclic, the diagram below shows the start of data reception.



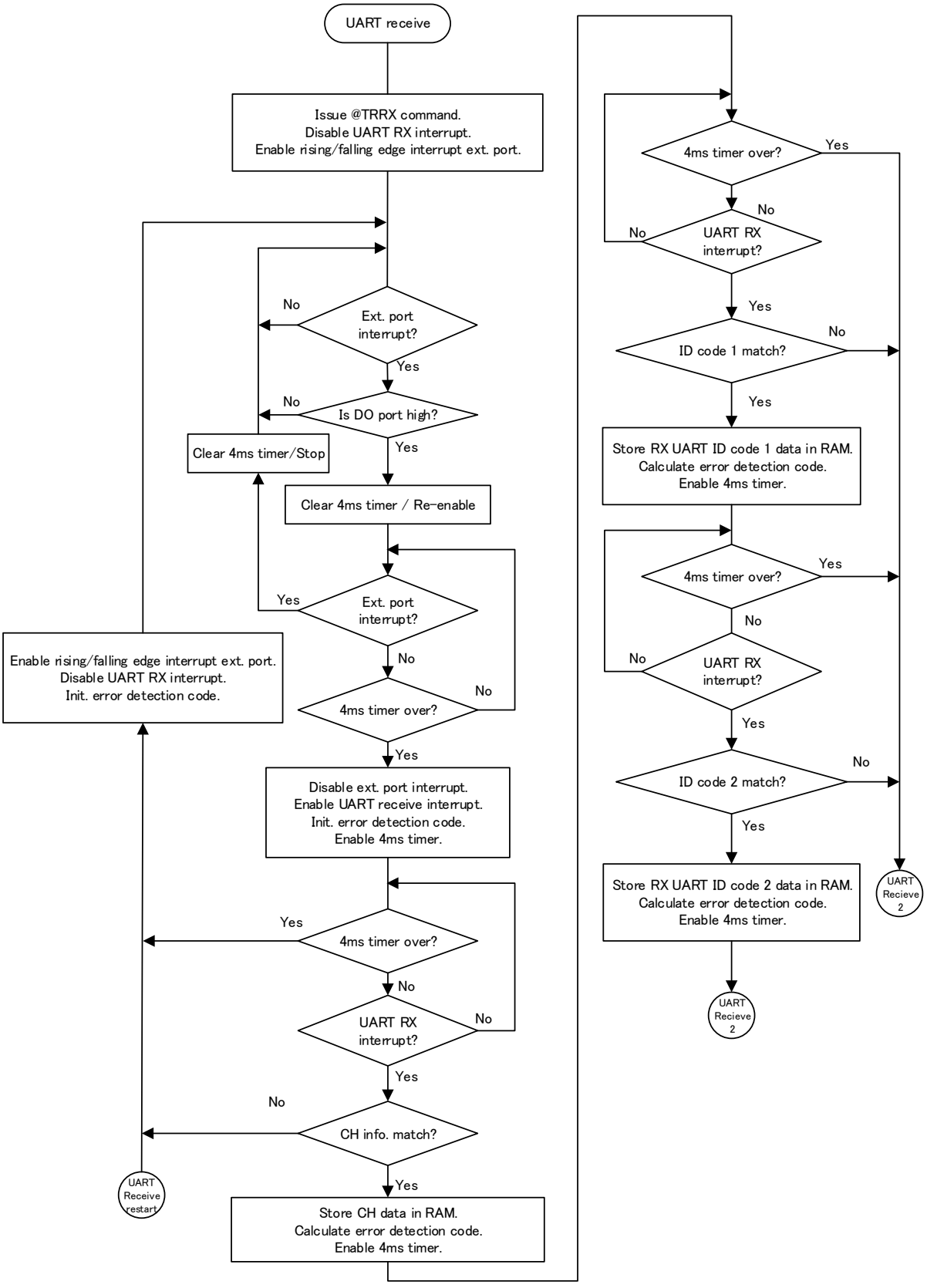
When reception starts in the middle of the cyclic transmission

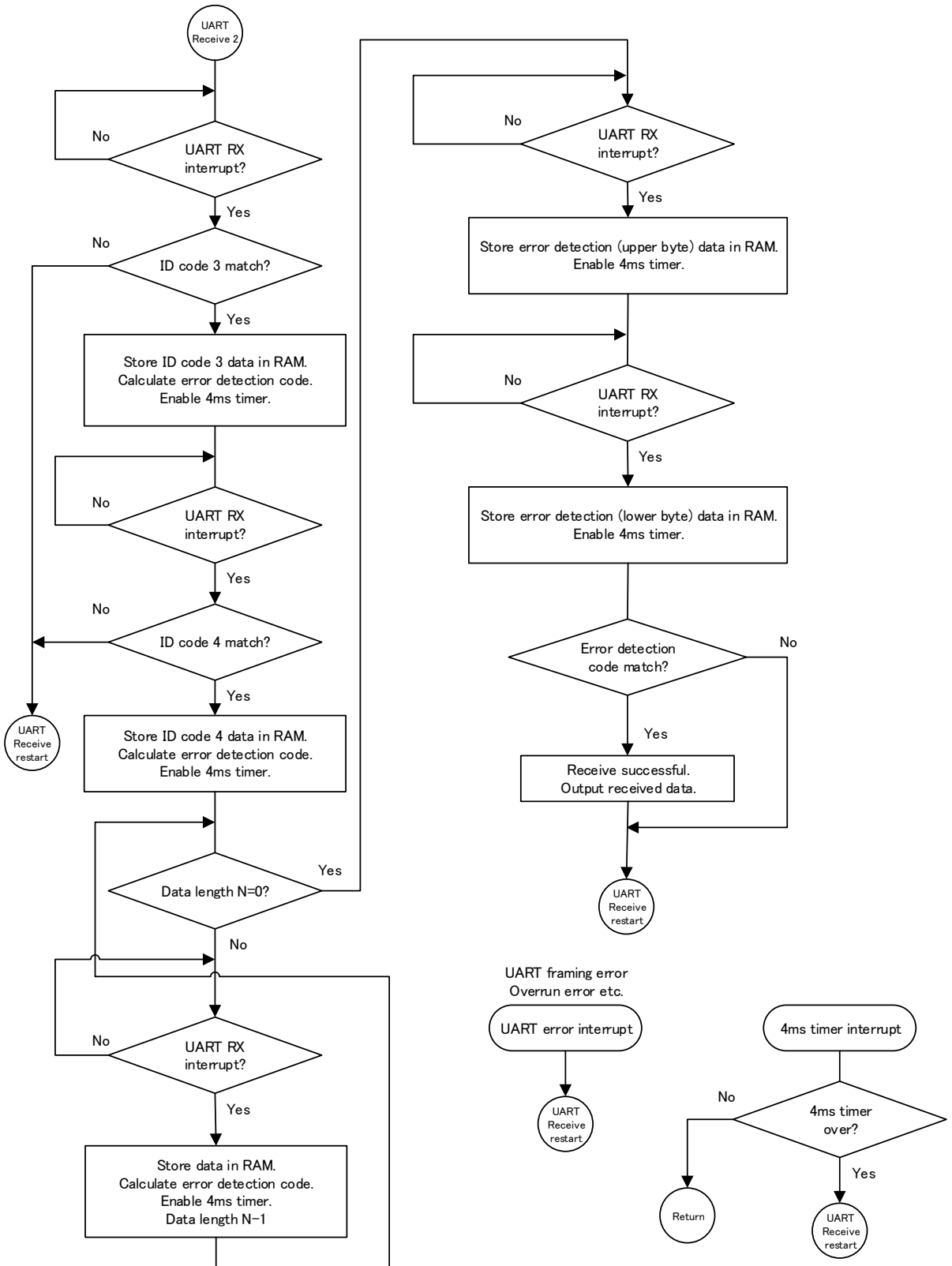
* Disable the monitoring of the 4ms HIGH (rising/falling edge interrupt) during data reception and transmission. If interrupt is enabled, detection of data cannot be done properly.

(Reference) Transmission using UART data in transparent mode



(Reference) Reception using UART data in transparent mode




UART Receive 2
No
UART RX interrupt?
Yes
No
ID code 3 match?
Yes
Store ID code 3 data in RAM.
Calculate error detection code.
Enable 4ms timer.
No
UART RX interrupt?
Yes
No
ID code 4 match?
Yes
UART Receive restart
Store ID code 4 data in RAM.
Calculate error detection code.
Enable 4ms timer.
Yes
Data length N=0?
No
No
UART RX interrupt?
Yes
Store data in RAM.
Calculate error detection code.
Enable 4ms timer.
Data length N-1
No
UART RX interrupt?
Yes
Store error detection (upper byte) data in RAM.
Enable 4ms timer.
No
UART RX interrupt?
Yes
Store error detection (lower byte) data in RAM.
Enable 4ms timer.
Error detection code match?
No
Yes
Receive successful.
Output received data.
UART Receive restart
UART framing error
Overrun error etc.
UART error interrupt
UART Receive restart
4ms timer interrupt
No
4ms timer over?
Yes
Return
UART Receive restart

Revision History

Issue	Date	Comment
1.0	2021/12/17	Start of document