IOB-UART, a RISC-V UART

Software User Guide, V0.1 , Build 9501dfb

May 25, 2022

1 Introduction

Software user guide for the IOb-UART software driver.

The present IOb-UART software drivers implement a way to interface with the IOb-UART peripheral for serial communication.

The present drivers provide base functionalities such as:

- initialization and setup
- basic control functions
- single character send and receive functions
- simple protocol for multi byte transfers

IOb-UART Defined Macros

- #define UART_PROGNAME "IOb-UART"
- #define STX 2
- #define ETX 3
- #define EOT 4
- #define ENQ 5
- #define ACK 6
- #define FTX 7
- #define FRX 8

IOb-UART Function Signatures

- void uart_init (int base_address, uint16_t div)

  Initialize UART.
- void uart_finish ()

  Close transmission.
- void uart_txwait ()

  Wait for TX.
- void uart_putc (char c)

  Print char.
- void uart_puts (const char ∗s)

  Print string.
- void uart_sendfile (char ∗file_name, int file_size, char ∗mem)

  Send file.
- void uart_rxwait ()

  Wait for RX Data.
- char uart_getc ()

  Get char.
- int uart_recvfile (char ∗file_name, char ∗∗mem)

  Receive file.

2 IOb-UART Macro Values

2.1 ACK

```
#define ACK 6
```

Acknowledge. Signal reception of incomming message.

2.2 ENQ

```
#define ENQ 5
```

Enquiry. Signal start of UART connection.

2.3 EOT

```
#define EOT 4
```

End of transmission. Signal end of UART connection.

2.4 ETX

```
#define ETX 3
```

End text. Signal end of data sequence to be printed.

2.5 FRX

```
#define FRX 8
```

File reception. Signal file reception request.

2.6 FTX

```
#define FTX 7
```

File transfer. Signal file transfer request.

2.7 STX

```
#define STX 2
```

Start text. Signal start of data sequence to be printed.

2.8 UART_PROGNAME

```
#define UART_PROGNAME "IOb-UART"
```

Prefix to IOb-Uart specific prints.

3 IOb-UART Functions

3.1 uart_finish()

```
void uart_finish ( )
```

Close transmission.

Send end of transmission (EOT) command via UART. Active wait until TX transfer is complete. Use this function to close console program.

Returns

void.

3.2 uart_getc()

```
char uart_getc ( )
```

Get char.

Active wait and receive char/byte from UART.

Returns

received byte from UART.

3.3 uart_init()

```
void uart_init (
              int base_address,
              uint16_t div )
```

Initialize UART.

Reset UART, set IOb-Uart base address and set the division factor. The division factor is the number of clock cycles per simbol transfered.

For example, for a case with fclk = 100 Mhz for a baudrate of 115200 we should have `div=(100*10^6/115200) = (868).`

The following code is a simple usage example:

```
#include "iob-uart.h"
#define UART_BASE (0x80000000)
#define FREQ (100000000)
#define BAUD (115200)
int main()
{
  //init uart
  uart_init(UART_BASE,FREQ/BAUD);
  uart_puts("\n\n\nHello world!\n\n\n");
  uart_finish();
}
```

The IOb-UART is inicialized with `UART_BASE` as the memory address and `div=(FREQ/BAUD).`

Parameters

base_address	IOb-Uart instance base address in the system.
div	Equal to round (fclk/baudrate).

Returns

void.

3.4 uart_putc()

```
void uart_putc (
              char c )
```

Print char.

Send character via UART to be printed by in console program.

Parameters

c	Character to print.

Returns

void.

3.5 uart_puts()

```
void uart_puts (
             const char * s )
```

Print string.

Send string via UART to be printed by in console program.

Parameters

s	Pointer to char array to be printed.

Returns

void.

3.6 uart_recvfile()

```
int uart_recvfile (
             char * file_name,
             char ** mem )
```

Receive file.

Request variable size file via UART. Order of commands:

1. Send file receive (FRX) command.
2. Send file_name.
3. Receive file_size (in little endian format).
4. Send ACK command.
5. Receive file.

If memory pointer is not inicialized, allocates memory for incomming file.

Parameters

file_name	Pointer to file name string.
mem	Pointer in memory to store incomming file.

Returns

Size of received file.

3.7 uart_rxwait()

```
void uart_rxwait ( )
```

Wait for RX Data.

Active wait for RX incomming data.

Returns

void.

3.8 uart_sendfile()

```
void uart_sendfile (
             char * file_name,
             int file_size,
             char * mem )
```

Send file.

Send variable size file via UART. Order of commands:

1. Send file transmit (FTX) comnand.
2. Send file_name.
3. Send file_size (in little endian format).
4. Send file.

Parameters

file_name	Pointer to file name string.
file_size	Size of file to be sent.
mem	Pointer to file.

Returns

void.

3.9 uart_txwait()

```
void uart_txwait ( )
```

Wait for TX.

Active wait until TX is ready to process new byte to send.

Returns

void.