UNC Holdings UK47XD RISC Processor Programming Manual - Sparknotes Version

Version 1.0

April 2025

Overview

The **UK47XD** is an 32bit microcontroller designed for secure embedded applications. It features a simple memory layout, UART-based communication, and built-in security features to prevent tampering and unauthorized code execution. It is primarily used in the lighting industry.

Memory Map

The memory map layout which programmers can access is organized as follows:

Address Range	Description
`0x0000-0x036F`	Boot Section
`0x0370-0x0EDBFF`	Application Flash (User Area)

> **Note**: Any access to any other addresses will result in a hard fault, requiring a reboot of the processor.

UART Communication

UK47XD communicates with external devices via UART at 115200 baud, 8N1.

UART Registers

Address	Name	Description
`0x0100`	`UART_DATA`	Data Register (R/W)
`0x0101`	`UART_STAT`	Status Register (R)
`0x0102`	`UART_CTRL`	Control Register (W)

UART Status Register Bits

- Bit 0: TX Ready
- Bit 1: RX Available
- Bit 7: Error Flag

Programming Mode

The UK47XD can be programmed over UART. In order to signal to the processor that you would like to interact with it over UART, simply send the bytes `[0x55, 0x0, 0xC1, 0x0]` within 5 seconds of booting the processor. The processor should then respond with an acknowledgement packet.

Programming Mode Commands

Packet Format The transport packet format for all transactions is as follows:

```
struct Packet {
    uint8_t HEAD; // 0x33
    uint16_t LEN; // Length of data that comes after this field, in little endian.
    PACKET_INTERNAL_DATA DATA; // Data that is stored here
}

struct PACKET_INTERNAL_DATA {
    COMMANDS CMD; // Command Index
    RESPONSES STATUS; // Status, used by the system whenever something goes wrong.
    uint8_t ARGS[LEN - 2]; // The bytes for the arguments. It will be LEN - 1.
}
```

Command Indices

```
typedef enum {
    UNKNOWN = 0x0,
    COMM_INIT = 0x3,
    SET_CHIP_FREQ = 0x5,
    ID_AUTHENTICATION = 0x34,
    READ = 0x69,
} COMMANDS;
```

Response/Status Indices

```
typedef enum {
    ACK = 0x50,
    INVALID_COMMAND = 0x80,
    FLOW_ERROR = 0x81,
    UNAUTHORIZED = 0x82,
    INVALID_FREQUENCY = 0x83,
    INVALID_ID_LEN = 0x84,
```

```
    INVALID_ADDRESS = 0x87,
    INVALID_ADDRESS_ALIGNMENT = 0x88,
} RESPONSES;
```

Responses The status field in a data packet can be set to one of the following values:

- ACK: Used as an acknowledgement.
- INVALID_COMMAND: Used to report a bad command.
- FLOW_ERROR: Used to report a command sent out of order.
- UNAUTHORIZED: Used to signify an authentication failure.
- INVALID_FREQUENCY: Used to signify an invalid frquency being set.
- INVALID_ID_LEN: Used to signify a malformed ID.
- INVALID_ADDRESS: Used to signify a bad/out of bounds address.
- INVALID_ADDRESS_ALIGNMENT: Used to signify an address that is not 0x400 aligned.

Data Packets As implied by the above, the inline data packets need a `type`, `status`, and `args` section (if applicable) for successful communication.

- The `type` will be of the command being sent or acknowledged.
- The `status` is one of those in the RESPONSE enum. If you are sending a command, you do not need to fill out this field; the UK47XD will fill this field, however.
- The `args` section just contains the raw data that you are trying to pass in.

Commands.COMM_INIT Initializes communication with the processor over UART. This must be sent before any other command is sent, otherwise you will receive a packet with a `FLOW_ERROR`. Upon success, you will receive an acknowledgement back from the system.

Example packet structure: `[0x33, 0x2, 0x0, 0x3, 0x0]`

Commands.SET_CHIP_FREQ Sets the processor speed in megahertz. This must be sent after the `COMM_INIT` command. Upon success, you will receive an acknowledgement back from the system. You must use either 8 or 16 MHz, as these are the only two operating frequencies.

Example packet structure (to set to 8 MHz): `[0x33, 0x5, 0x0, 0x5, 0x00, 0x12, 0x7A, 0x00]`

Commands.ID_AUTHENTICATION Unlocks the chip for reading. This must be sent after the `COMM_INIT` command, and the `SET_CHIP_FREQ`, in that order. Upon success, you will receive an acknowledgement back from the system. This cannot be bruteforced; if an attempt is made, the chip is wiped for security reasons (see below).

Example packet structure (assuming the password is DEADBEEFDEADBEEF): `[0x33, 0x11, 0x0, 0x34, 0x44, 0x43, 0x41, 0x44, 0x42, 0x45, 0x45, 0x46, 0x44, 0x43, 0x41, 0x44, 0x42, 0x45, 0x45, 0x46]`

Commands.READ Reads 0x400 bytes from a region. This must be sent after the unlocking the chip (see `ID_AUTHENTICATION`). Upon success, you will receive an acknowledgement back from the system, containing the 0x400 bytes.

Example packet structure (reading from 0x112233): `[0x33, 0x5, 0x0, 0x69, 0x33, 0x22, 0x11, 0x00]`

Security Features

The UK47XD includes several mechanisms for runtime and firmware security.

- An ID can be used to secure your processor. This is a 16 byte ID consisting of CAPITAL alphabetical characters 'A' through 'Z'. If a user attempts to bruteforce this password, the chip will be erased, preserving the contents.
- The JTAG fuses can be blown to prevent tampering, or the OTP register set.
- SWD is disabled unless you specifically request a development board from UNC Holdings.
- As stated before, the bootrom and other secure assets cannot be read out, only user area.
- Sudden voltage changes or glitching attempts will trigger a system reset.