MT7681 TCP IP

agenda

- Introduction
- Feature support
- uIP main control loop
- Memory management
- API introduction
- Programming introduction
  - General description
  - How to add a UDP APP
  - How to add a TCP APP
- Programming introduction
  - How to use DHCP and static IP
  - How to use DNS

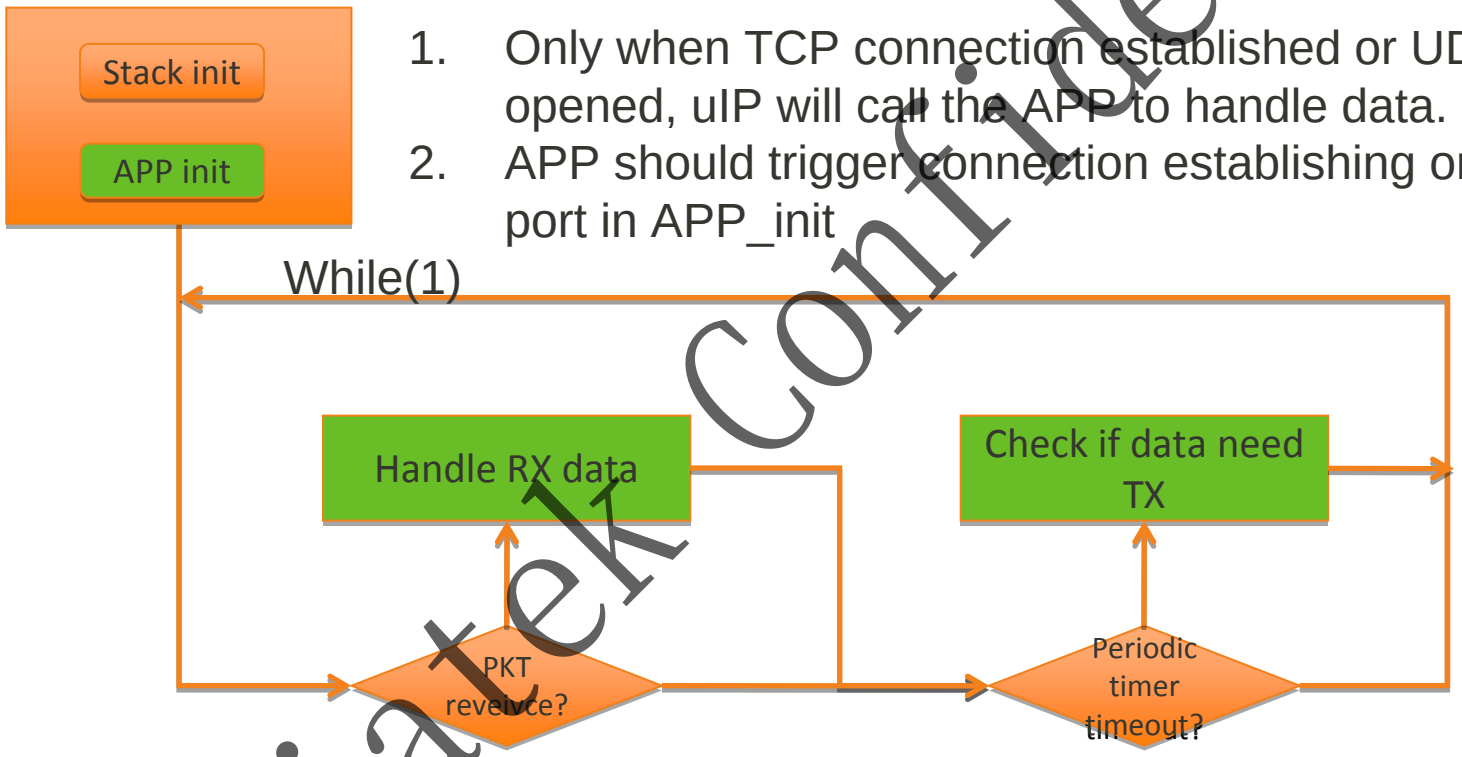
uIP introduction

- uIP introduction
  - The uIP TCP/IP stack is intended to make it possible to communicate using the TCP/IP protocol suite even on small 8-bit micro-controllers.
  - The uIP implementation is designed to have only the absolute minimal set of features needed for a full TCP/IP stack.

Features support

features	Supported by uIP	Supported on MT7681
ARP	Y	Y
ICMP	Y	Y
IP	Y	Y
TCP	Y	Y
UDP	Y	Y
DHCP and static IP	Y	Y
DNS	Y	Y
Web client/server/telnet/smtp	Y	N

Main control loop



1. Only when TCP connection established or UDP local port opened, uIP will call the APP to handle data.
2. APP should trigger connection establishing or open local port in APP_init

uIP memory management

- u8_t uip_buf[UIP_BUFSIZE+2]
  - The uIP stack does not use explicit dynamic memory allocation. Instead, it uses a(only one) single global buffer for holding packets.
- struct uip_conn uip_conns[UIP_CONNS]
  - Fixed table for holding TCP connections
- struct uip_udp_conn uip_udp_conns[UIP_UDP_CONNS]
  - Fixed table for holding UDP “connections”.

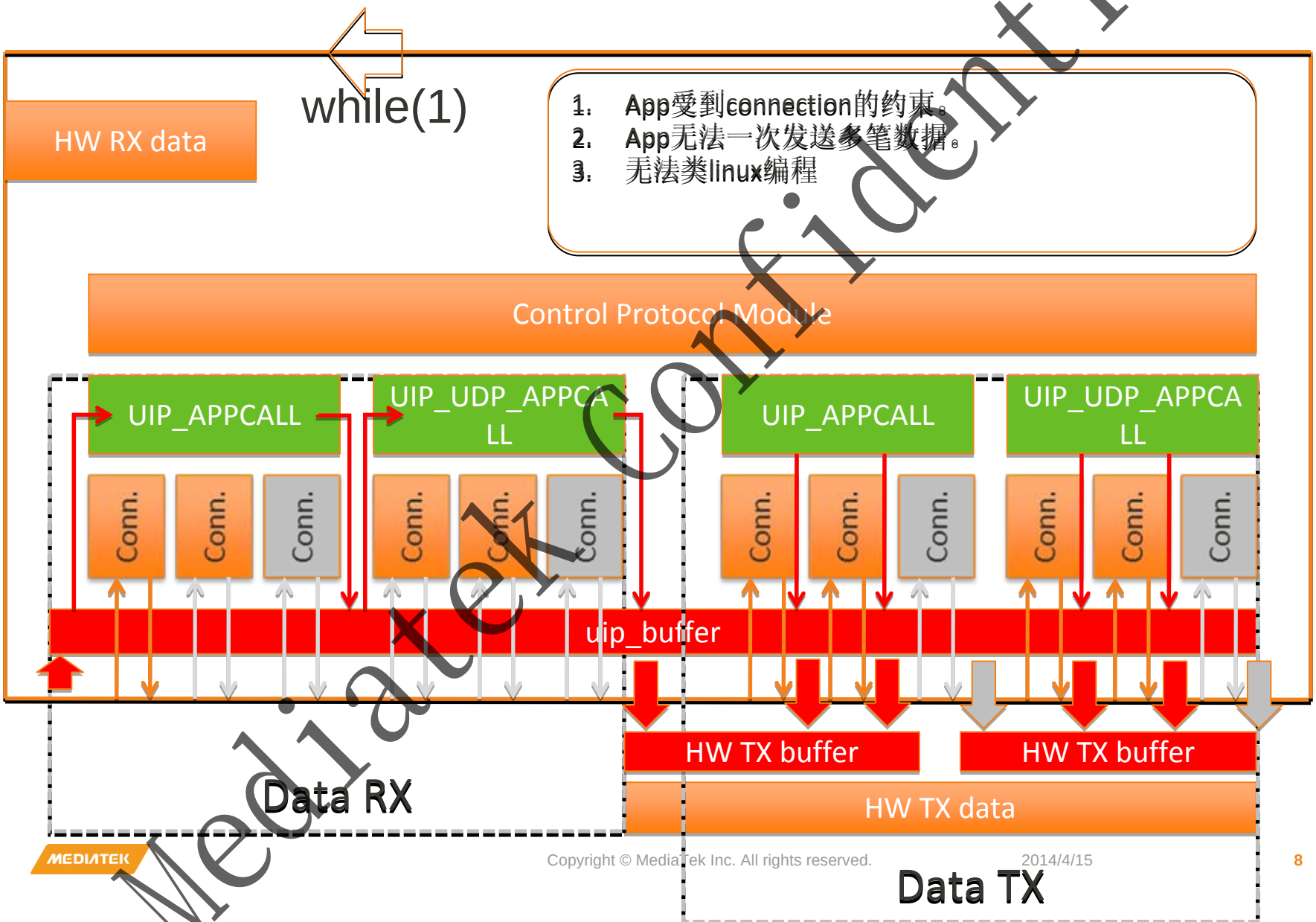
API introduction

- uIP didn’t use the BSD socket API because:
  - The BSD socket API use stop-and-wait semantics and it need multitasking OS support.
  - MT7681 running in s single task and don’t have OS
  - The overhead of task management, context switching and allocation of stack space for the tasks might be too high in the intended MT7681 architectures

API introduction

- uIP API uses an event driven interface where the application is invoked in response to certain events, those event including:
  - data is received
  - data has been successfully delivered
  - a new connection has been set up
  - when data has to be retransmitted
  - Etc..

Event driven interface.



Receiving data

- uIP will call UIP_APPCALL to inform the APP new data arrived.
  - uip_newdata()
    - Check if new data arrived.
  - uip_datalen()
    - The length of the data
  - uip_appdata
    - A global pointer pointed to the data buffer.

Sending data

- App send data by using uip_send()
- uIP don’t support TCP retransmit by default, instead, it need APP to handle rexmit event.
- uIP on MT7681 support TCP retransmit when turn on the option CFG_SUPPORT_TC_REXMIT
- The application can send only one chunk of data at a time on a connection and it is not possible to call uip_send() more than once per application invocation; only the data from the last call will be sent.

Open connection

- For TCP Client
  - uip_connection()
  - After connection establish, uIP will call UIP_APPCALL, APP can call test function uip_connected to check whether a new connection established.
- For TCP server
  - uip_listen()

Open connection

- For UDP
  - uip_udp_new()
    - Assign a udp “connection”
  - uip_udp_bind()
    - Bind to a local port

Example code

- Please refer to the source code

Source code introduction

- MT7681 TCPIP include 4 folders
  - apps
    - TCPIP APP include DHCP, DNS, TCP/UDP client server example code
  - lib
    - Necessary library
  - mt76xx
    - TCPIP main function and HW TX/RX interface
  - uip
    - Stack core