1. Developement Environment Building

1 Download Contiki Source Code (under Linux):

Firstly, clone the Contiki repository:

git clone https://github.com/contiki-os/contiki .git

Secondly, use the following command to update the firmware and every dependency about cc26xx(sensrortag):

Thirdly, Under the just downloaded Contiki repository file and give in the following command:

git sudmodule --init -recursive

2 Update the Debugger DevPack firmware (under windows):

Firstly, download the XDS Emulation Software Package and install it.

Secondly, under Windows using the bash command to get into the right path:

```
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation。保留所有权利。

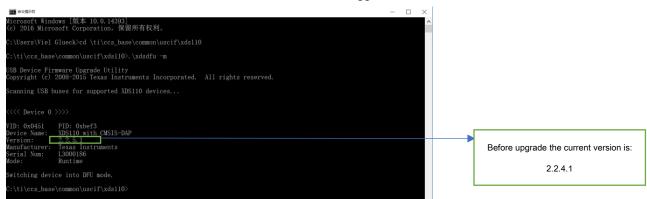
C:\Users\Viel Glueck>cd \ti\ccs_base\common\uscif\xds110

C:\ti\ccs_base\common\uscif\xds110>
```

Thirdly, use the following command:

.\ xdsdfu -m

This command is used to see the firmware version of the Debugger.



Fourthly, using the following command to upgrade it:

.\xdsdfu -f firmware.bin -r

.\xdsdfu -m

```
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation。保留所有权利。

C:\Users\Viel Glueck\cd \ti\ccs_base\common\uscif\xds110

C:\ti\ccs_base\common\uscif\xds110\cdot\ xdsdfu -m

USB Device Firmware Ugrade Utility
Copyright (c) 2008-2015 Texas Instruments Incorporated. All rights reserved.

Scanning USB buses for supported XDS110 devices...

\( \text{VID: 0x0451} \quad \text{PID: 0xbef3} \\
Device \text{Now161} \quad \text{PID: 0xbef3} \\
Device \text{Nome: XDS110 with CMSIS-DAP Version: 2.2.5.1 \\
Manufacturer: Texas Instruments \\
Serial \text{Num: Banutime Serial Num: L3000186 \\
Mode: Runtime

Switching device into DFU mode.

C:\ti\ccs_base\common\uscif\xds110\cdot\
C:\ti\ccs_base\common\uscif\xds110\cdot\
```

Now the current version is 2.2.5.1, the upgrade is done!

Note: When upgrade the debugger must be disconnected from sensortag!

3 A simple RPL-UDP example:

In this example two cc2650 sensortag will build a UDP connection using RPL protocol. The client will send 'Hello' to the server. We can see the result through the UART on the Debugger to see the Result.

Firstly, we compile the source code under Linux. Go to the right path of the example:

cd /Desktop/contiki/example/ipv6/rpl-udp

Then we build the server code and the client code separately: Take client for example:

sudo make TARGET=srf06-cc26xx BOARD=sensrotag/cc2650 udp-client.bin

Before we compile the server code, we must clean the middle files of the previous project:

sudo make TARGET=srf06-cc26xx BOARD=sensrotag/cc2650 udp-client.bin clean

Then we can compile the server code:

sudo make TARGET=srf06-cc26xx BOARD=sensrotag/cc2650 udp-server.bin

We can find the output files are:

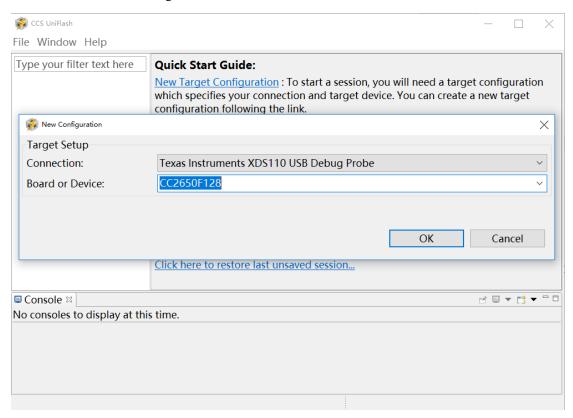
udp-client.bin

udp-client.elf

udp-client.bin

udp-client.elf

Secondly, we need to flash it into our sensortag. It is recommended to flash it under Windows OS. We need to download the Uniflash and install it. When open the Uniflash software, we create a new file and set as following:

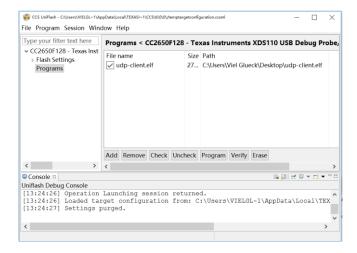


Select on **Flash Setting** menu and click on **Erase Entire Flash.** <u>It'll tell you that you are not running the latest version and it needs to be upgrade. Just click the upgrade and wait.</u>

```
| C22:55:09| Begin Erase Entire Flash operation. | C22:56:00| Cortex_M3_0: MassErase(): Initializing. | C22:56:00| Cortex_M3_0: MassErase(): Issuing Board Reset. | C22:56:00| Cortex_M3_0: MassErase(): Mass erase complete. | C22:56:00| Operation Erase Entire Flash returned. | C22:56:00| Operation Erase Entire Flash Returned Entire
```

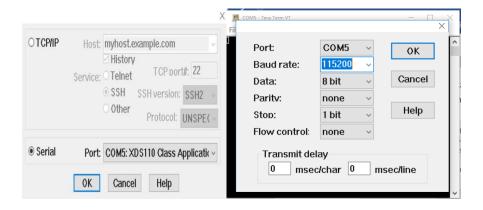
The above console output tells you it is been upgraded.

Thirdly, flash it with your code: click programs table and add .elf file, click program.



4 Debugger results:

Firstly, it is recommended to use an open source software called <u>Tera Term</u> to monitor the com port on your computer. Note: The Baud rate of the UART is 115200.



Secondly. We can see the client is sending hello to server:

```
COM100:115200baud - Tera Term VT

File Edit Setup Control Window Help

Starting Contriki-3.x-3155-ga26ee64

With DriverLib v0.47020

II CC2650 LaunchPad

IEEE 802.15.4: Ves. Sub-GHz: No. BLE: Ves. Prop: Ves

Net: sicslowpan

MRC: CSHIR

RDC: nullrdc

RF: Channel 25

Node ID: 40576

UDP client Pvo6 addresses: fd00::212:4b00:7b1:9e80

fe80::212:4b00:7b1-9e80

Created a connection with the server :: local/remote port 8765/5678

DRIR send to 1 'Hello 1'

DRIR send to 1 'Hello 2'

This is udp-client

This is udp-client

COM100:115200baud - Tera Term VT

File Edit Setup Control Window Help

Starting Contiki-3.x-3155-ga26ee64

With DriverLib v0.47020

II CC2650 LaunchPad

IEEE 802.121:4.6: ves. Sub-GHz: No. BLE: Yes. Prop: Yes

Net: sicslowpan

MRC: CSHIR

RDC: nullrdc

RF: Channel 25

Node ID: 65413

UDP server started. nbr:10 routes:10

created a new RPL dag

Server IPv6 addresses: fd00::212:4b00:b24:ff85

fd00::ff:fe00:1

fe08::212:4b00:b24:ff85

Created a server connection with remote address :: local/remote port 8768/8765

DRIR recv 'Hello 1 from the client' from 128* AIR recv 'Hello 2 from the client' from 128* AIR recv 'Hello 2 from the client' from 128*

This is udp-server
```

Hardware AES: 2.

According to the latest updated cc26xxware, there is a ROM AES API:

```
AES_ECB_EncryptData(uint8_t *text, uint16_t textLen, uint8_t *aesKey);
```

which can be called by the user application. The discussion on the TI forum shows that, there is a initial process before the above API is called, this initial process is allocating a absolute RAM scratch area for the ROM AES stack remapping. This remapping is done at:

https://github.com/contiki-os/cc26xxware/tree/master/utils/rom_crypto

According the code document, it should be called before the API usage. So my code is like:

```
StandaloneRomCryptoInit();
AES_ECB_EncryptData(uint8_t *text, uint16_t textLen, uint8_t *aesKey);
and also there is a strange symbol in the standalone_rom_crypto.c:
commonROMScratchArea[53]@0x20004F2C;
i guess this is for specific compiler, so i change this to:
                                          ((section(".COMMENROMSCRATCHAREA")))\\
uint32_t
                   __attribute__
commonROMScratchArea[53];
here the .COMMENROMSCRATCHAREA refers to the absolute address: 0x20004F2C
the according linkerfile has been added .COMMENROMSCRATCHAREA like followings:
.COMMENROMSCRATCHAREA 0x20000754:
     {
    KEEP(*(.COMMENROMSCRATCHAREA))
     } > SRAM
after all this after compiling contiki, the c2650 is stuck at the API:
```

The test application is rpl-udp in directory Contiki/example/ipv6/rpl-udp. One senortag uses the ROM aes to send encrypted data to another sensortag.

AES_ECB_EncryptData(uint8_t *text, uint16_t textLen, uint8_t *aesKey);

However this AES part seems not working.