ISP Programming Guide

Application Note for 8-bit NuMicro® Family

Document Information

Abstract	This application note describes the boot code architecture, I/O interfaces, including UART, I^{2}C or RS485 and the usage of ISP commands.
Apply to	NuMicro® MS51 / ML51 / ML54 / ML56 series.

Table of Contents

1 Overview

The 8-bit NuMicro® Family microcontroller (MCU) supports In-System-Programming (ISP) function allowing the embedded flash memory to be reprogrammed under the control of Nuvoton's NuMicro® ISP Programming Tool. ISP is performed to upgrade firmware through on-chip connectivity interface, such as UART, I²C, and RS485, without removing the microcontroller from the system.

Nuvoton provides the customized total solution of ISP programming application. Depending on the product feature, the Host may be a PC, a microcontroller or other devices. To develop ISP application, Nuvoton provides the ISP programming software tool including source code and execution file.

In terms of product firmware upgrade of ISP, Nuvoton provides resources integral from program development to firmware upgrade. For this purpose, Nuvoton provides the ISP programming tool including source code and software utility for host and sample code in the NuMicro® Board Support Package (BSP) for device.

After download and unzip ISP programming tool from website, the directory information is listed in Table 1-1.

Directory	Description
Application Program	Software utility with source code and the boot code of MCU.
User Manual	This document describes the user manual of ISP programming tool

Table 1-1 The ISP Programming Tool Directory Information

To upgrade product firmware by ISP programming application, the Nu-Link2-Pro has the built-in communication bridge of UART, I²C, SPI, CAN or RS-485 interface. Therefore, users can do the firmware upgrade without purchasing any other serial communication bridges. In addition, users can use the corresponding ISP programs in the NuMicro® Board Support Package (BSP) without developing programs by themselves, which can save the development time. The functions Nu-Link2-Pro supports are fully documented and the requirements are specified.

Item	Description
BSP	● Board Support Package(BSP) for MS51 Series ● Board Support Package(BSP) for ML51 Series
Tool	● ICP Programmer Tool V3.05.7174r or later ● ISP Programmer Tool V4.00 or later ● Nu-Link Keil Driver V3.05.7174 or later ● Nu-Link IAR Driver V3.05.7174 or later

1.1 Introduction

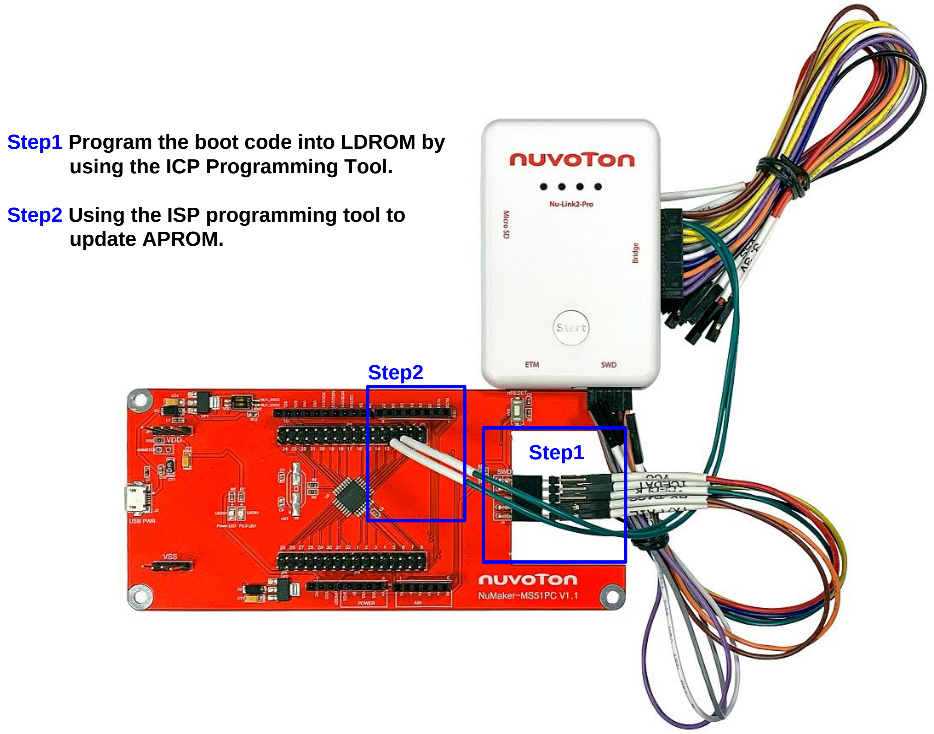
In-System-Programming (ISP) makes the product easy to update, whether the product is under development or in the hands of an end user. ISP performs Flash Memory updating without removing the microcontroller from the system. It allows a device to be re-programmed under the software's control. Furthermore, knowing the way of updating the tool's firmware, users can implement a wider range of ISP application.

There are 2 steps to implement ISP:

1. Program the boot code into LDROM by using a universal programmer or Nuvoton's NuMicro® ICP Programming Tool, and configure the CBS(CONFIG0[7]) to boot from LDROM. The connection is shown as Figure 1-1. Please be noted that the maximum size of LDROM is 4K Byte
2. Using the ISP programming tool to update APROM or CONFIG, and the connection is shown as Figure 1-1.

Take the MS51 16KB series as an example: the Program Memory stores the program codes to execute as shown in Figure 1-2.

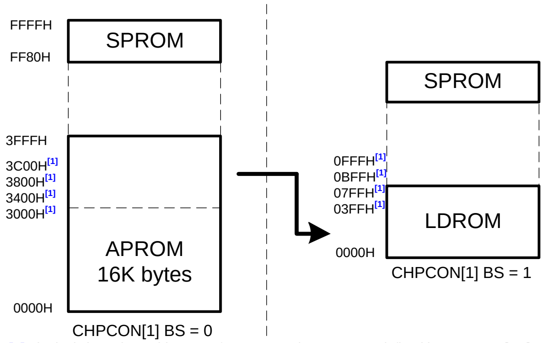
In general, ISP programming tool transfers the new application code to MCU through multi-interfaces by programming the boot code into LDROM. Then MCU receives the new application code and re-programs it into APROM by IAP commands. Nuvoton provides ISP firmware and ISP programming tool for the 8-bit NuMicro® Family microcontroller. For more details, please refer to Nuvoton 80C51 Microcontroller Technical Support.



Step1 Program the boot code into LDROM by using the ICP Programming Tool.

Step2 Using the ISP programming tool to update APROM.

Figure 1-1 ISP programming Steps



[1] The logic boundary addresses of APROM and LDROM are defined by CONFIG1[2:0].

Figure 1-2 Program Memory Map and Boot Select

2 The In-System-Programming Interface

This chapter introduces the pin definition and the description of I/O interfaces, such as UART, I²C, and RS485. And this chapter will also introduce the method to connect the Nu-Link2-Pro to a computer, and the way to connect a development board or a product to Nu-Link2-Pro via an individual interface. For more details, please refer to Nu-Link2-Pro Technical Support.

2.1 The 8-bit NuMicro® Family Pin Definition and Function Connection

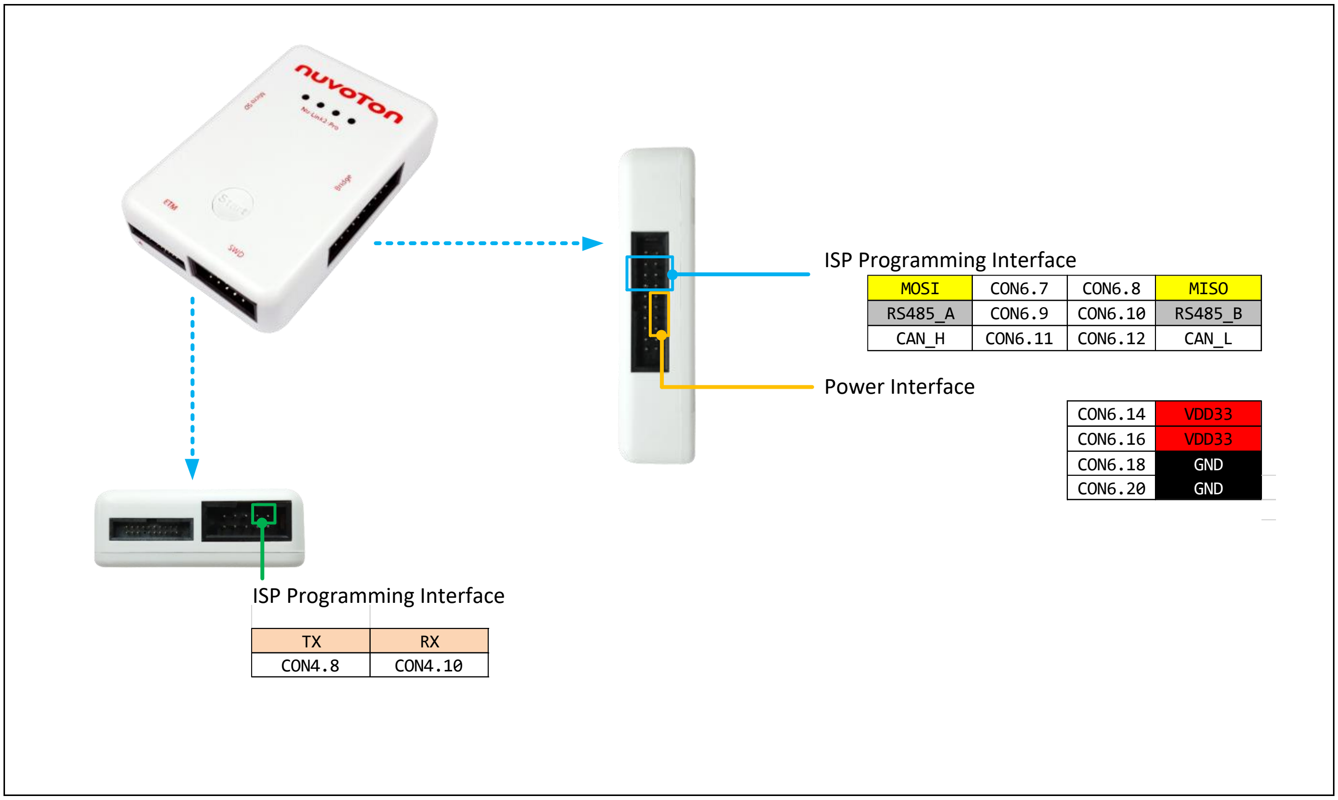
Table 2-1 shows pins definition for each interface.

ISP Interface / Chips	UART				RS485		I²C	
	UART0		UART1					
	TX	RX	TX	RX	TX	RX	SCL	SDA
MS51BA9AE / MS51DA9AE	P0.6	P2.0	P1.6	P1.2	P1.6	P1.2	P0.3	P0.4
MS51FB9AE / MS51XB9AE	P0.6	P0.7	P0.2	P1.6	P0.2	P1.6	P1.3	P1.4
MS51FC0AE / MS51XC0BE / MS51EC0AE / MS51TC0AE / MS51PC0AE	P0.6	P0.7	P0.2	P1.6	P0.2	P1.6	P1.3	P1.4
ML51 / ML54 / ML56 series	P3.1	P3.0	P2.3	P2.2	P2.3	P2.2	P2.5	P2.4

Table 2-1 The 8-bit NuMicro® Family Pin Definition for ISP Programming Interface

2.2 Nu-Link2-Pro Bridge Interface Pin Definition and Function Connection

Figure 2-1 shows the Nu-Link2-Pro pin definition of each interface. Users can freely select a suitable interface to program.



ISP Programming Interface

MOSI	CON6.7	CON6.8	MISO
RS485_A	CON6.9	CON6.10	RS485_B
CAN_H	CON6.11	CON6.12	CAN_L

Power Interface

CON6.14	VDD33
CON6.16	VDD33
CON6.18	GND
CON6.20	GND

ISP Programming Interface

TX	RX
CON4.8	CON4.10

Figure 2-1 Pin Definition of Nu-Link2-Pro Connectors

2.2.1 UART Connection

The UART connection pins are VDD33 (CON6.14 and CON6.16), VSS (CON6.18 and CON6.20), TX (CON4.8) and RX (CON4.10). Figure 2-2 presents how to connect the target board to use UART function and Table 2-2 shows the pin corresponding to the target board.

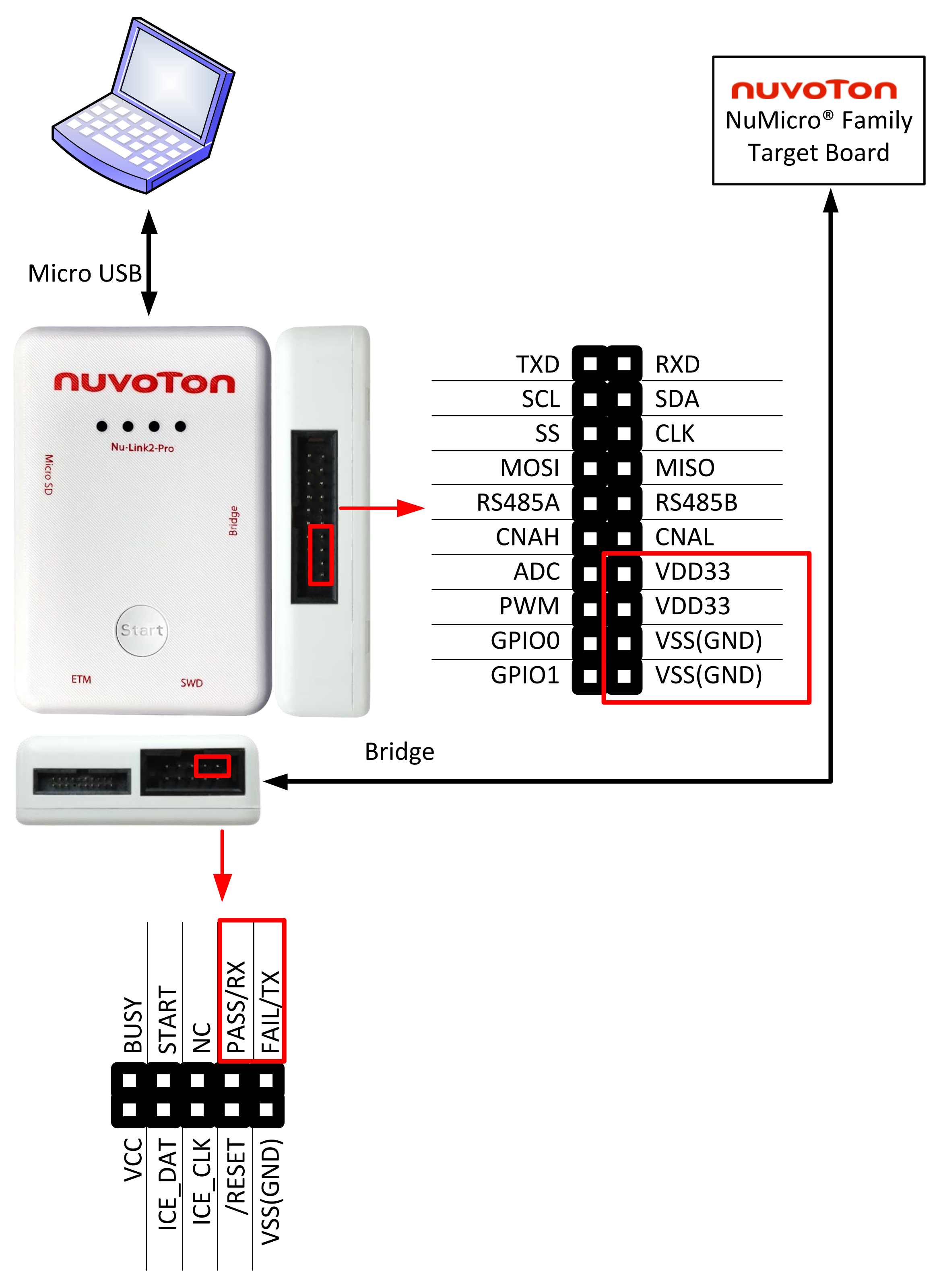


Figure 2-2 Bridge Interface Connection Diagram for UART

<table>
<tr><th>Pin Name</th><th>Pin Number</th><th>Pin Corresponding to the Target Board</th></tr>
<tr><td>FAIL/TX</td><td>CON4.8</td><td>RXD[1]</td></tr>
<tr><td>PASS/RX</td><td>CON4.10</td><td>TXD[1]</td></tr>
<tr><td>VDD33</td><td>CON6.14</td><td>VDD[1]</td></tr>
<tr><td>VDD33</td><td>CON6.16</td><td>VDD[1]</td></tr>
<tr><td>VSS(GND)</td><td>CON6.18</td><td>VSS(GND)</td></tr>
<tr><td>VSS(GND)</td><td>CON6.20</td><td>VSS(GND)</td></tr>
<tr><td colspan="3"><b>Note:</b>
1. The target board power and signal only supports 3.3V at Nu-Link2-Pro Bridge interface.
2. The UART settings are as follows:
◆ Baud Rate:115200bps
◆ Data: 8 bit
◆ Parity: None
◆ Stop bits: 1 bit</td></tr>
</table>

Table 2-2 Bridge Interface Corresponding Pin for UART

2.2.2 I^{2}C Connection

The I^{2}C connection pins are VDD33 (CON6.14 and CON6.16), VSS (CON6.18 and CON6.20), SCL (CON6.3) and SDA (CON6.4). Figure 2-3 presents how to connect the target board to use I^{2}C function and Table 2-3 shows the pin corresponding to the target board.

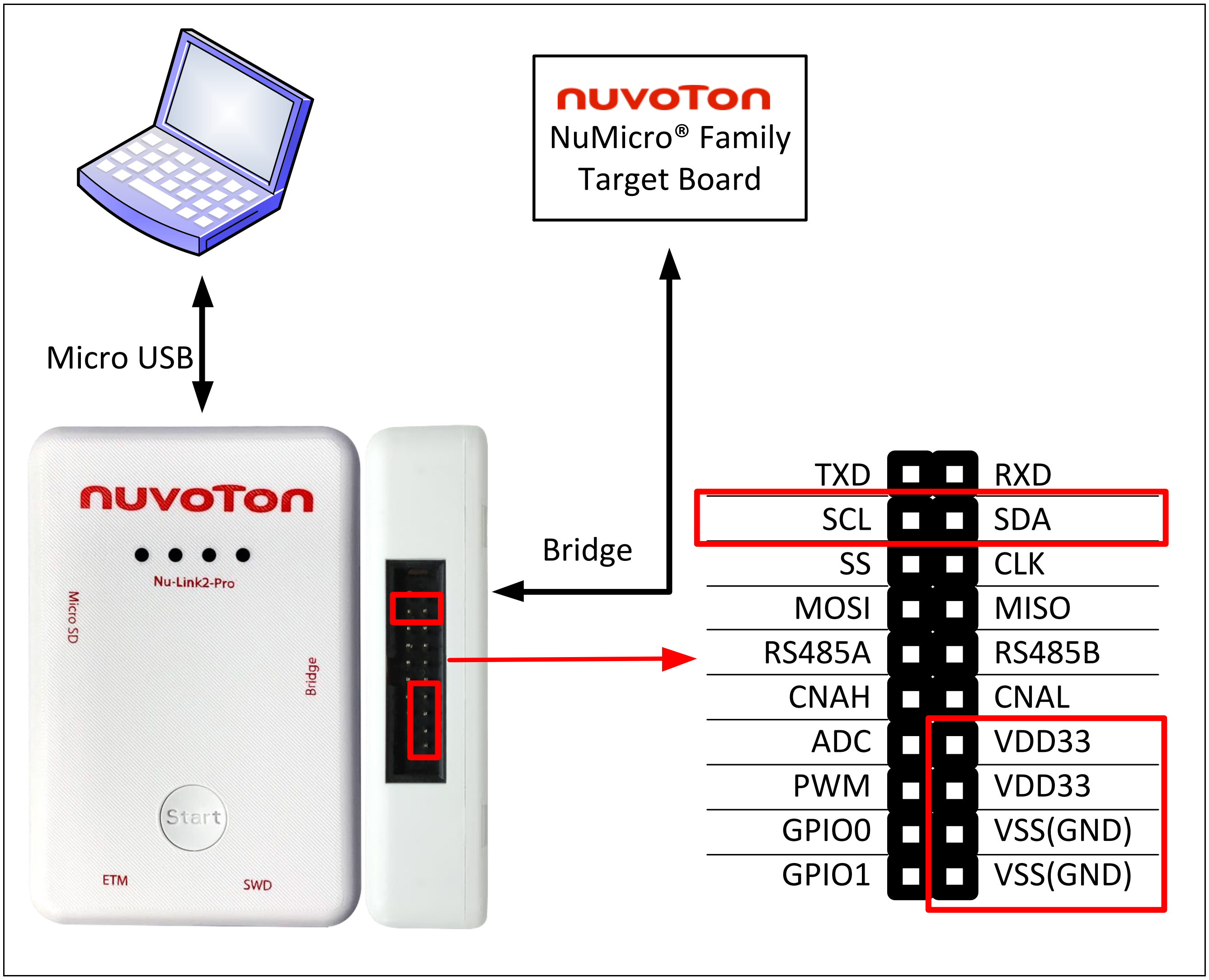


Figure 2-3 Bridge Interface Connection Diagram for I^{2}C

Pin Name	Pin Number	Pin Corresponding to the Target Board
SCL[1]	CON6.3	SCL[2]
SDA[1]	CON6.4	SDA[2]
VDD33	CON6.14	VDD[2]
VDD33	CON6.16	VDD[2]
VSS(GND)	CON6.18	VSS(GND)
VSS(GND)	CON6.20	VSS(GND)

Note:

1. Internal 4.7 kΩ pull-up resistors R67 and R68 on Nu-Link2-Pro; the user can adjust them according to needs.
2. The target board power and signal only support 3.3 V at Nu-Link2-Pro Bridge interface.

Table 2-3 Bridge Interface Pin for I^{2}C

2.2.3 RS-485 Connection

The RS-485 connection pins are VDD33 (CON6.14 and CON6.16), VSS (CON6.18 and CON6.20), RS485A (CON6.9) and RS485B (CON6.10). Figure 2-4 presents how to connect the target board to use RS-485 function and Table 2-4 shows the pin corresponding to the target board.

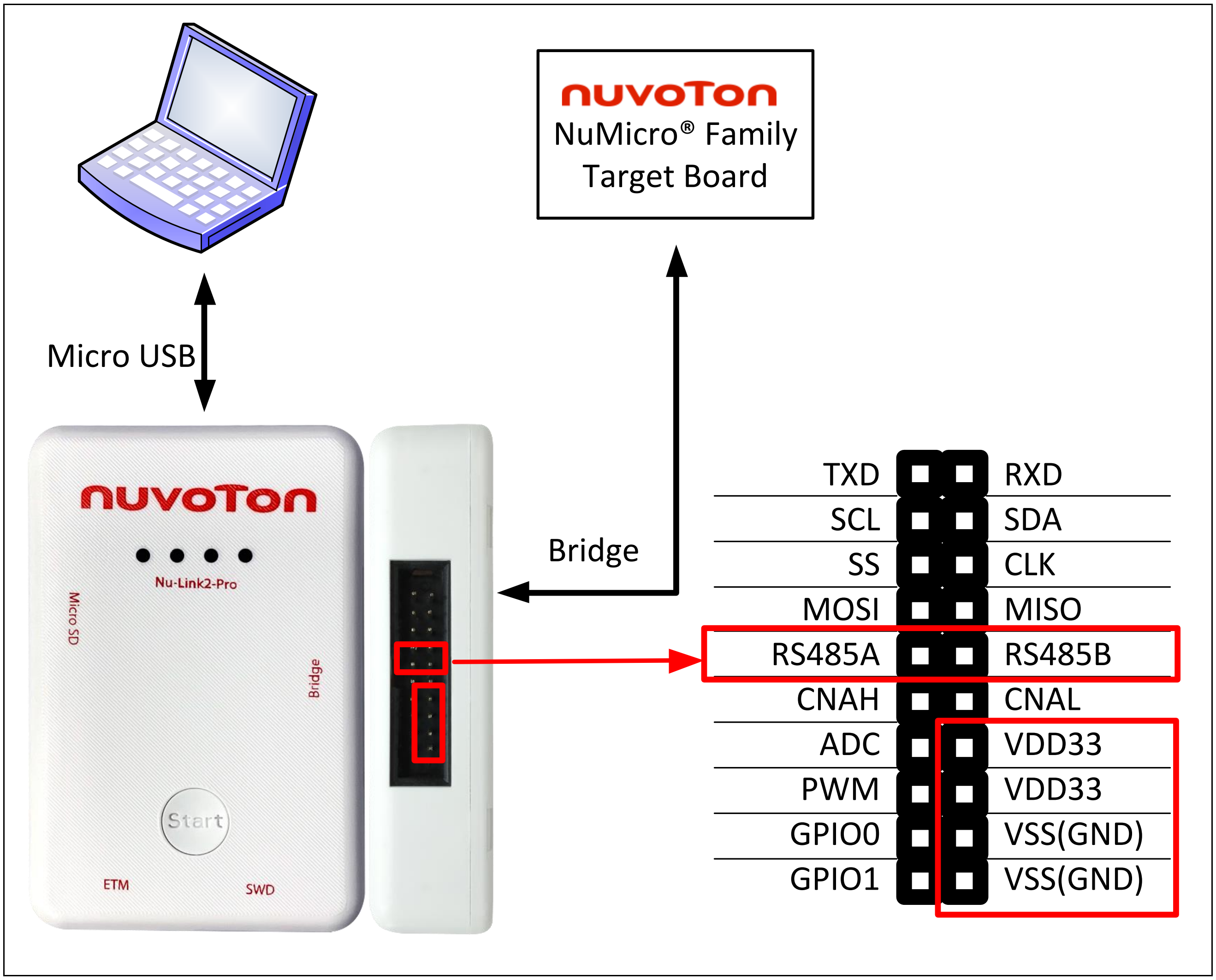


Figure 2-4 Bridge Interface Connection Diagram for RS-485

Pin Name	Pin Number	Pin Corresponding to the Target Board
RS485A[1]	CON6.9	RS485A[2]
RS485B[1]	CON6.10	RS485B[2]
VDD33	CON6.14	VDD[2]
VDD33	CON6.16	VDD[2]
VSS(GND)	CON6.18	VSS(GND)
VSS(GND)	CON6.20	VSS(GND)

Note:

1. Internal 120 Ω terminal resistors R62 on Nu-Link2-Pro; the user can adjust them according to needs.
2. The target board power and signal only support 3.3 V at Nu-Link2-Pro Bridge interface.

Table 2-4 Bridge Interface Corresponding Pin for RS-485

3 In-System-Programming Commands

When 8-bit NuMicro® Family microcontroller receives a packet from ISP programming tool, it will parse the packet and take an action according to the transmitted command code. All commands have a common format consisting of 64 bytes. If users want to send the data less than 64 bytes, please pad it with leading zeros. The commands supported by ISP are listed in Table 3-1.

The MCU goes into a command parsing loop once it detects that an event condition occurs during ISP programming. The ISP program recursively checks the commands received from UART, I^2C or RS485, and then do the corresponding operations. Please reference to the Figure 3-1 ISP commands flow.

Command	Command Code	ACK	Command Description
CMD_UPDATE_APROM	0x000000A0	Yes	Within the range 0~MAX_APROM_SIZE, program APROM from address user specified.
CMD_UPDATE_CONFIG	0x000000A1	Yes	Write CONFIG0 and CONFIG1 registers of Flash memory.
CMD_READ_CONFIG	0x000000A2	Yes	Get CONFIG0 and CONFIG1
CMD_ERASE_ALL	0x000000A3	Yes	Erase all APROM, including Data Flash in Flash memory and CONFIG area. The CONFIG registers to restored to default value.
CMD_SYNC_PACKNO	0x000000A4	Yes	Synchronize packet number with NuMicro® Family microcontrollers before a valid command send.
CMD_GET_FWVER	0x000000A6	Yes	Get version information of ISP firmware.
CMD_GET_DEVICEID	0x000000B1	Yes	Get chip product ID.
CMD_RUN_APROM	0x000000AB	No	Instruct ISP firmware to boot from APROM.
CMD_RUN_LDROM	0x000000AC	No	Instruct ISP firmware to boot from LDROM.
CMD_CONNECT	0x000000AE	Yes	Test whether or not the ISP firmware is active.

Table 3-1 ISP Command Set

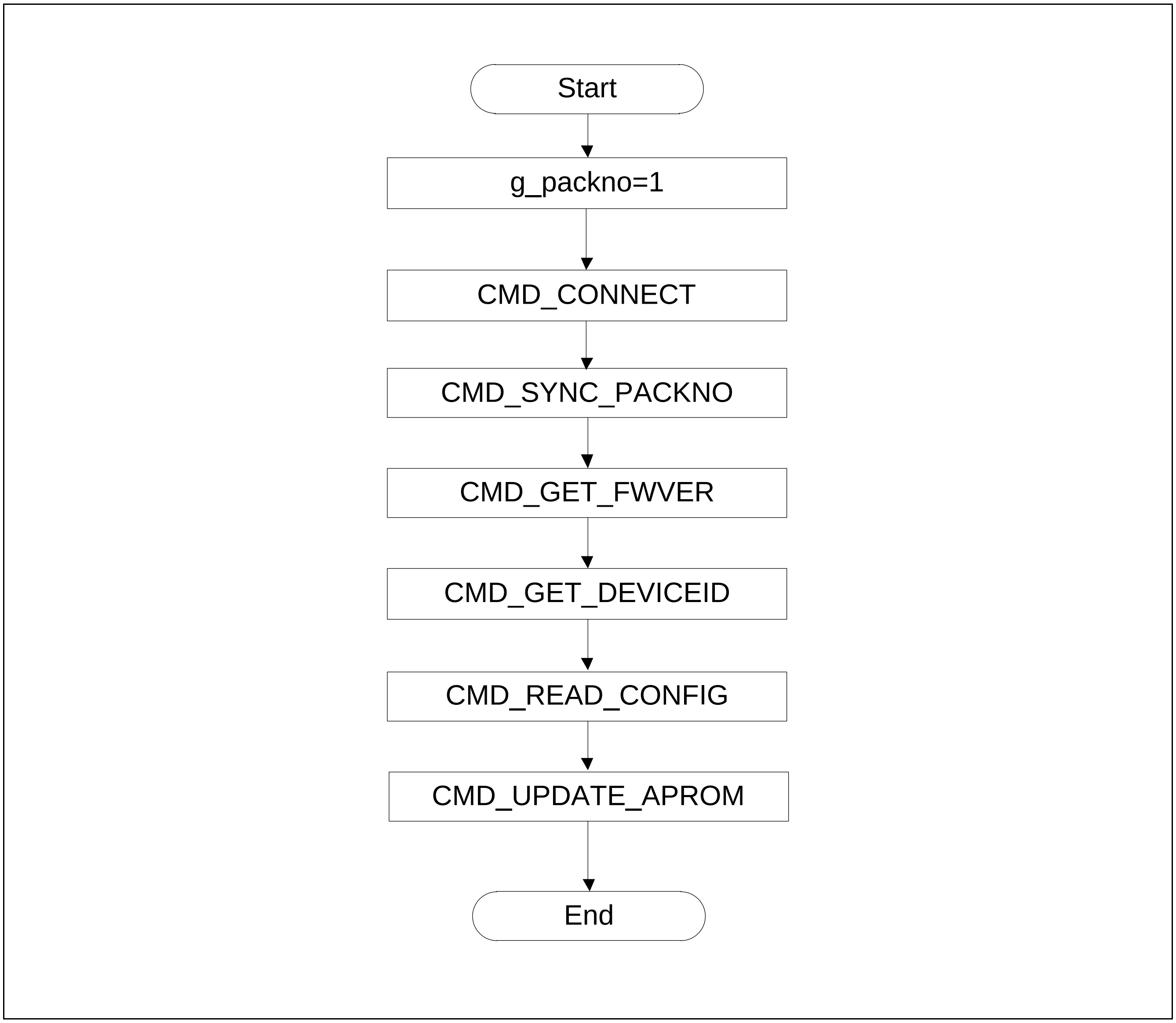


Figure 3-1 ISP Command Flow

3.1 CMD_UPDATE_APROM Command

This command is used to instruct the device to program APROM flash. Users can specify the start address and the length to be programmed. If the data is too big to be transferred in a packet, users can split the data into several packets and do the multiple transfers.

When the device receives the command packet, it erases all contents in APROM flash except for data flash, and reads the program data from received packet, and then the device programs the data into the specified address of flash memory. After the programming operation is completed, the device calculates the checksum of the whole package received, then transmits ACK packet to the host.

After ACK packet is received, host will check whether the checksum is right or not. If the checksum is right, it indicates that the data transfer is successful. After the operation is finished, users can continue to run another different operation or transmit another data packet.

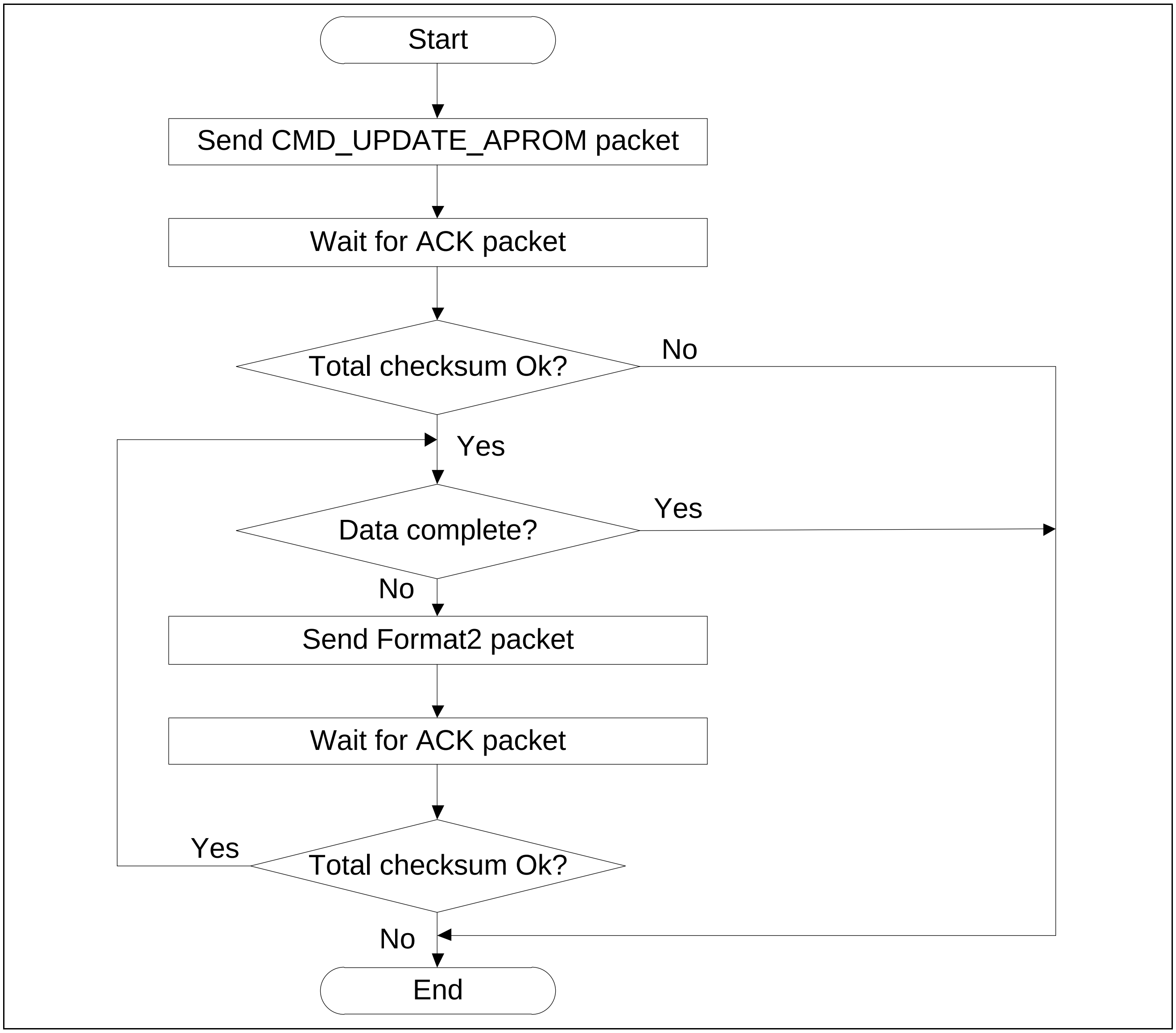


Figure 3-2 The Flow of Host Update APROM

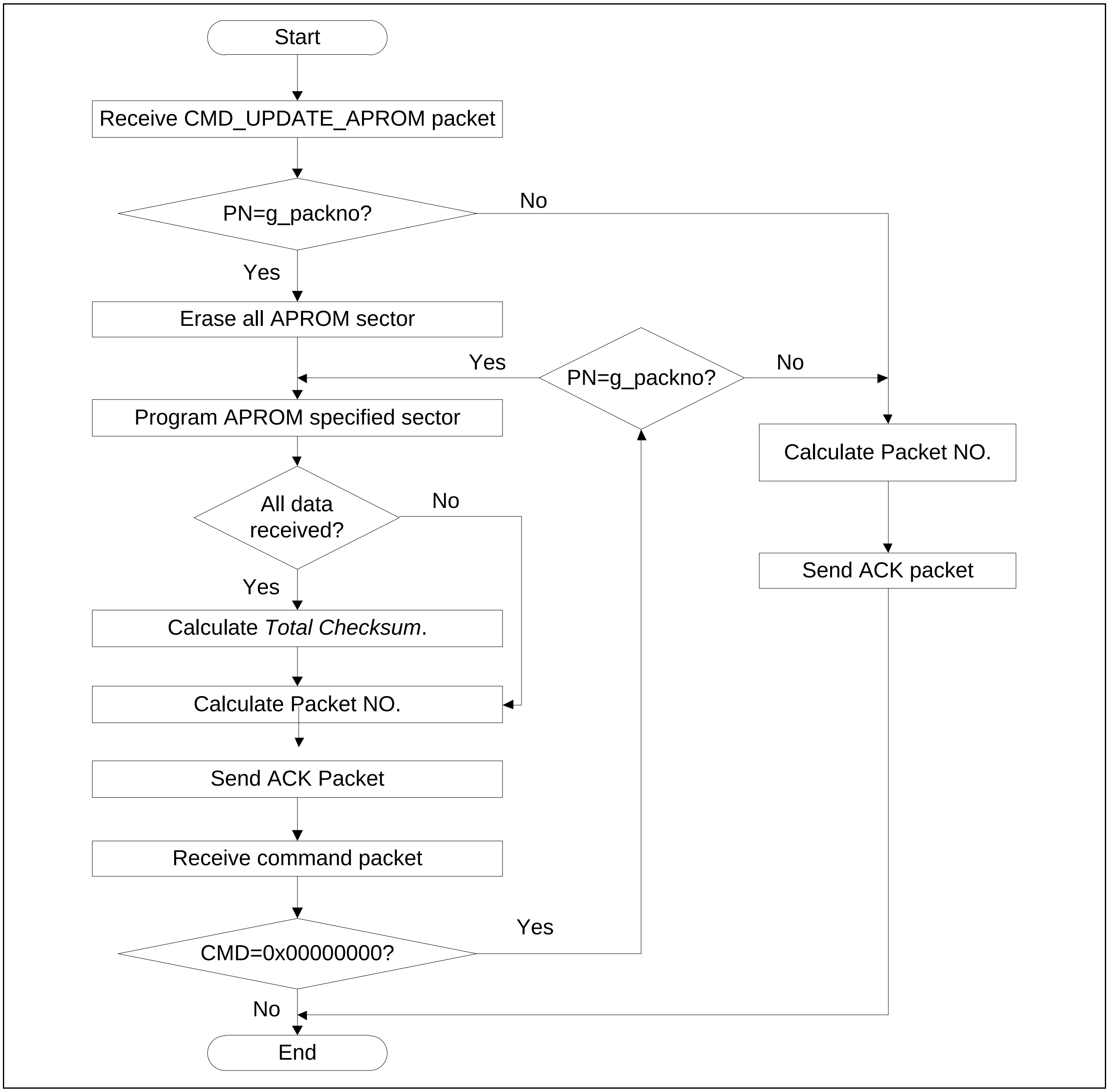


Figure 3-3 The Flow of Device Update APROM

3.1.1 Host Request Packet Format – 1

63	62	61	60	59	58	57	56
Data(47~40)							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**

Data(39~32)							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
Data(31~24)							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
Data(23~16)							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Data(15~8)							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Data(7~0)							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Total Length				Start Address			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:16]	**Data**	The value of binary or hex file to be written to the APROM, and the address is specified in Start Address.
[15:12]	**Total Length**	Total file length.
[11:8]	**Start Address**	Specify the address of data written in APROM.
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000A0

3.1.2 Host Request Packet Format – 2

If data can’t put into one packet, the packet format is as follows:

<table>
<tr><th>63</th><th>62</th><th>61</th><th>60</th><th>59</th><th>58</th><th>57</th><th>56</th></tr>
<tr><td colspan="8">Data(103~96)</td></tr>
<tr><th>55</th><th>54</th><th>53</th><th>52</th><th>51</th><th>50</th><th>49</th><th>48</th></tr>
<tr><td colspan="8">Data(95~88)</td></tr>
<tr><th>47</th><th>46</th><th>45</th><th>44</th><th>43</th><th>42</th><th>41</th><th>40</th></tr>
<tr><td colspan="8">Data(87~80)</td></tr>
<tr><th>39</th><th>38</th><th>37</th><th>36</th><th>35</th><th>34</th><th>33</th><th>32</th></tr>
<tr><td colspan="8">Data(79~72)</td></tr>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Data(71~64)</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">Data(63~56)</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">Data(55~48)</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="4">Packet Number</td><td colspan="4">0</td></tr>
</table>

Byte	Name	Description
[63:8]	**Data**	The value of binary or hex file to be written to the APROM, and the address is specified in Start Address.
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Reserved**	Reserved

3.1.3 Device Replay Packet Format

<table>
<tr><th>63</th><th>62</th><th>61</th><th>60</th><th>59</th><th>58</th><th>57</th><th>56</th></tr>
<tr><td colspan="8">0</td></tr>
<tr><th>55</th><th>54</th><th>53</th><th>52</th><th>51</th><th>50</th><th>49</th><th>48</th></tr>
</table>

0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				Total Package Checksum			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Total Package Checksum**	Sum up all bytes of this package.

3.2 CMD_UPDATE_CONFIG Command

This command is used to instruct the device to update CONFIG0 and CONFIG1. When the device receives the command packet, host reads the CONFIG0 and CONFIG1 data from the packet received and programs them into CONFIG area. Then the device reads the data from CONFIG area and fills them into ACK packet. After this, the device calculates the checksum of the whole packet received, then transmits ACK packet to the host.

After ACK packet is received, host will check whether the checksum is right or not. If the checksum is right, it indicates that the data transfer is successful. After the operation is finished, users can continue to run another different operation.

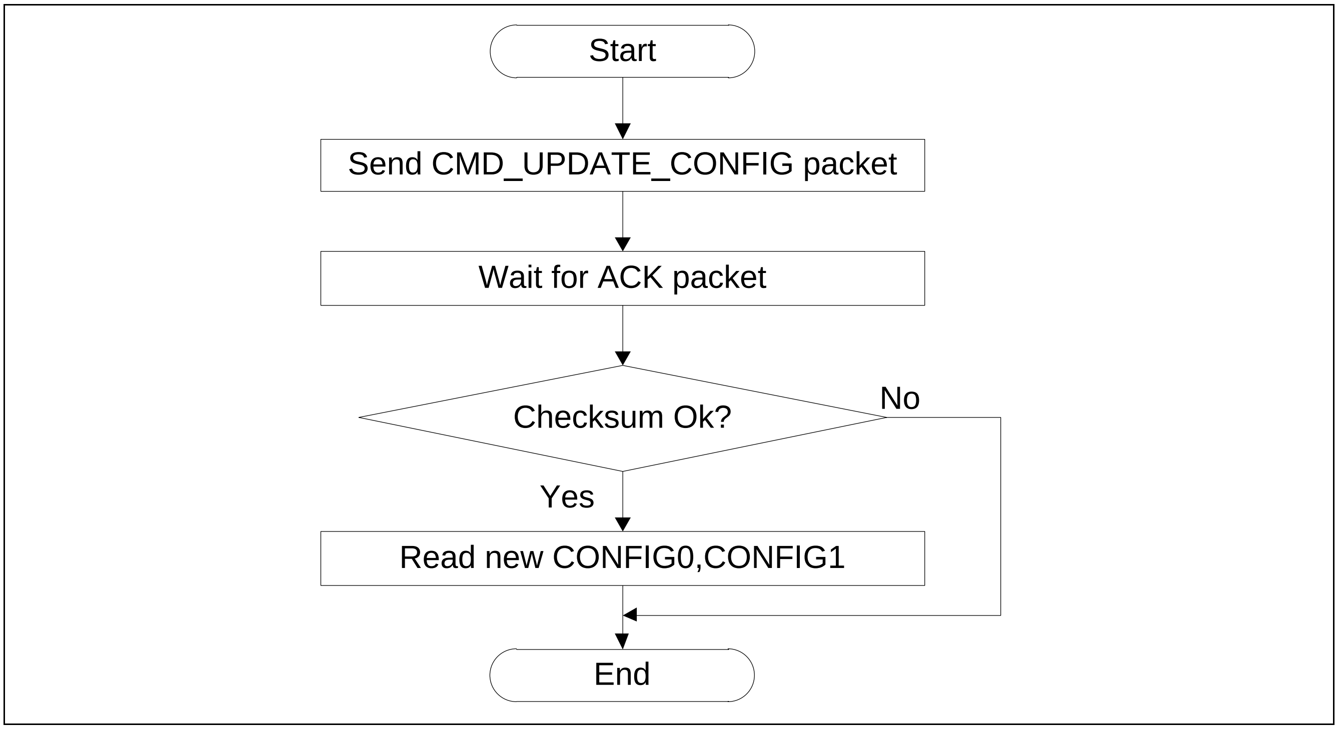
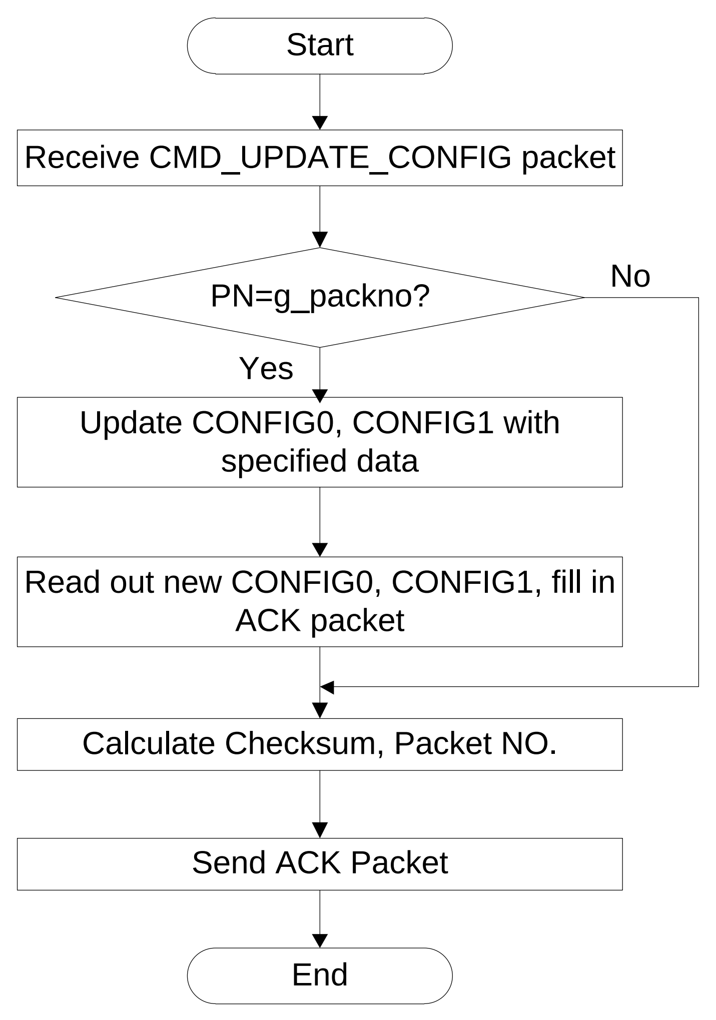


Figure 3-4 The Flow of Host Update Config0 and Config1


Start
Receive CMD_UPDATE_CONFIG packet
PN=g_packno?
No
Yes
Update CONFIG0, CONFIG1 with specified data
Read out new CONFIG0, CONFIG1, fill in ACK packet
Calculate Checksum, Packet NO.
Send ACK Packet
End

Note: PN is packet number received from host；g_packno is device local packet number.

Figure 3-5 The Flow of Device Update Config0 and Config1

3.2.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**

0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Config1				Config0			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:16]	**Reserved**	Reserved
[15:12]	**Config1**	The value of CONFIG1.
[11:8]	**Config0**	The value of CONFIG0.
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000A1

3.2.2 Device Replay Packet Format

63	**62**	**61**	**60**	**59**	**58**	**57**	**56**
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							

23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
New Config1				New Config0			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				Checksum			

Byte	Name	Description
[63:16]	**Reserved**	Reserved
[15:12]	**New Config1**	The value of new CONFIG1.
[11:8]	**New Config0**	The value of new CONFIG0.
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Checksum**	Sum up byte 0 to 11 of the received package.

3.3 CMD_READ_CONFIG Command Format

This command is used to instruct the device to read Config0 and Config1 of flash memory.

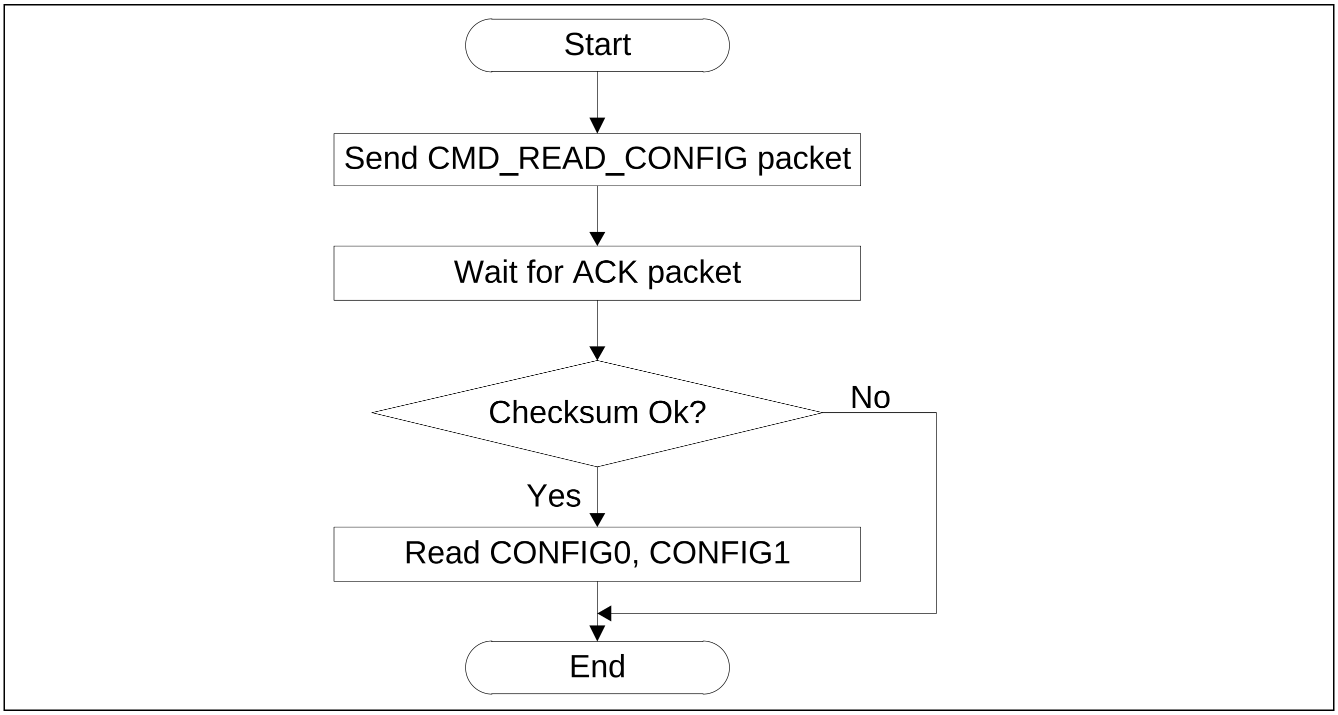


Figure 3-6 The Flow of Host Read Config0 and Config1

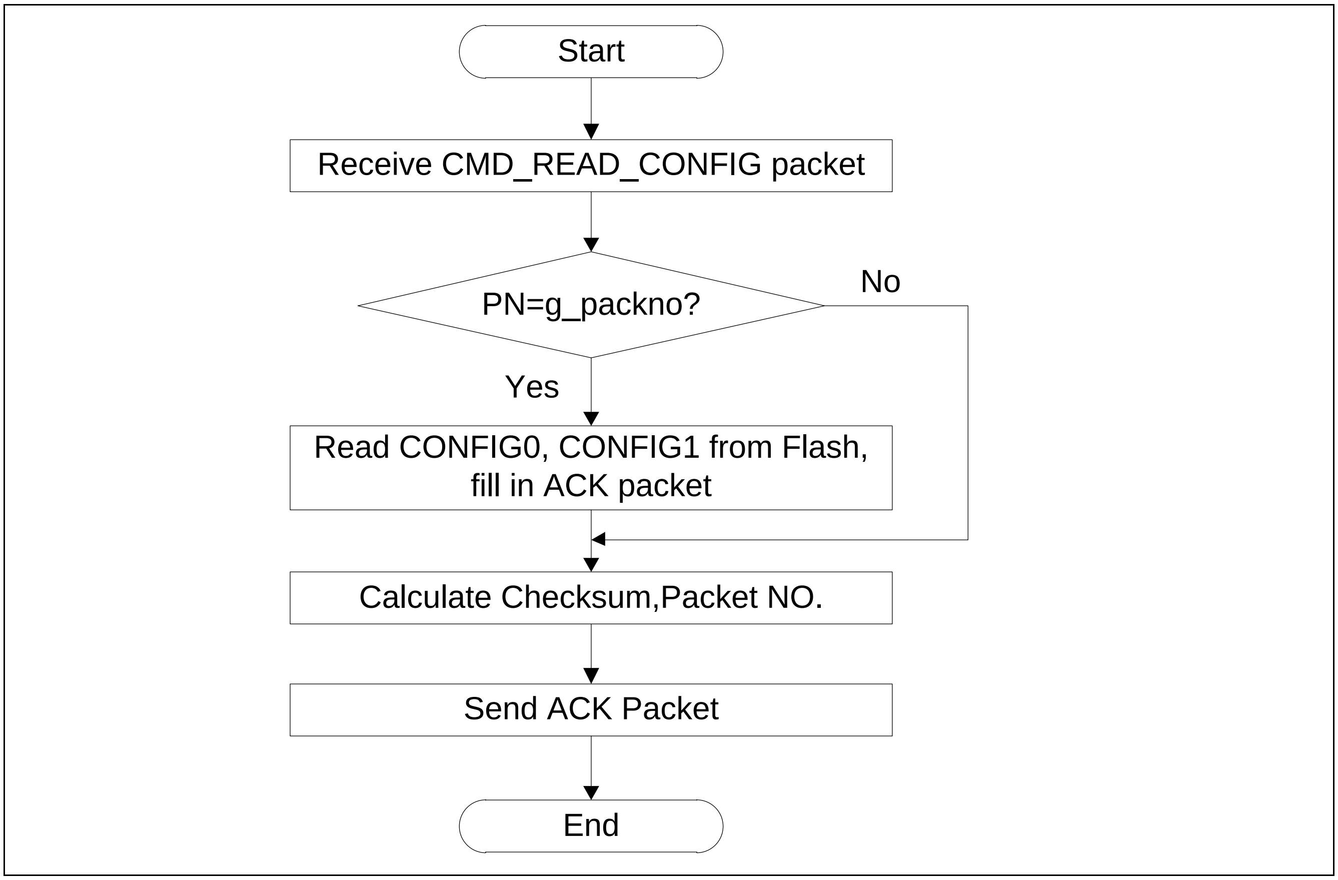

Start
Receive CMD_READ_CONFIG packet
PN=g_packno?
No
Yes
Read CONFIG0, CONFIG1 from Flash, fill in ACK packet
Calculate Checksum,Packet NO.
Send ACK Packet
End

Figure 3-7 The Flow of Device Read Config0 and Config1

3.3.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**

0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000A2

3.3.1 Device Replay Packet Format

63	**62**	**61**	**60**	**59**	**58**	**57**	**56**
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Config1				Config0			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				Checksum			

Byte	Name	Description
[63:16]	**Reserved**	Reserved
[15:12]	**Config1**	The value of CONFIG1.
[11:8]	**Config0**	The value of CONFIG0.
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Checksum**	Sum up byte 0 to 11 of the receive package.

3.4 CMD_ERASE_ALL Command Format

This command is used to instruct the device to erase APROM and Data Flash, and set CONFIG0 and CONFIG1 as the default value. The command format and the data transfer format are described as preceding section. When the device receives the command packet, it erases APROM, Data Flash, CONFIG0 and CONFIG1 registers.

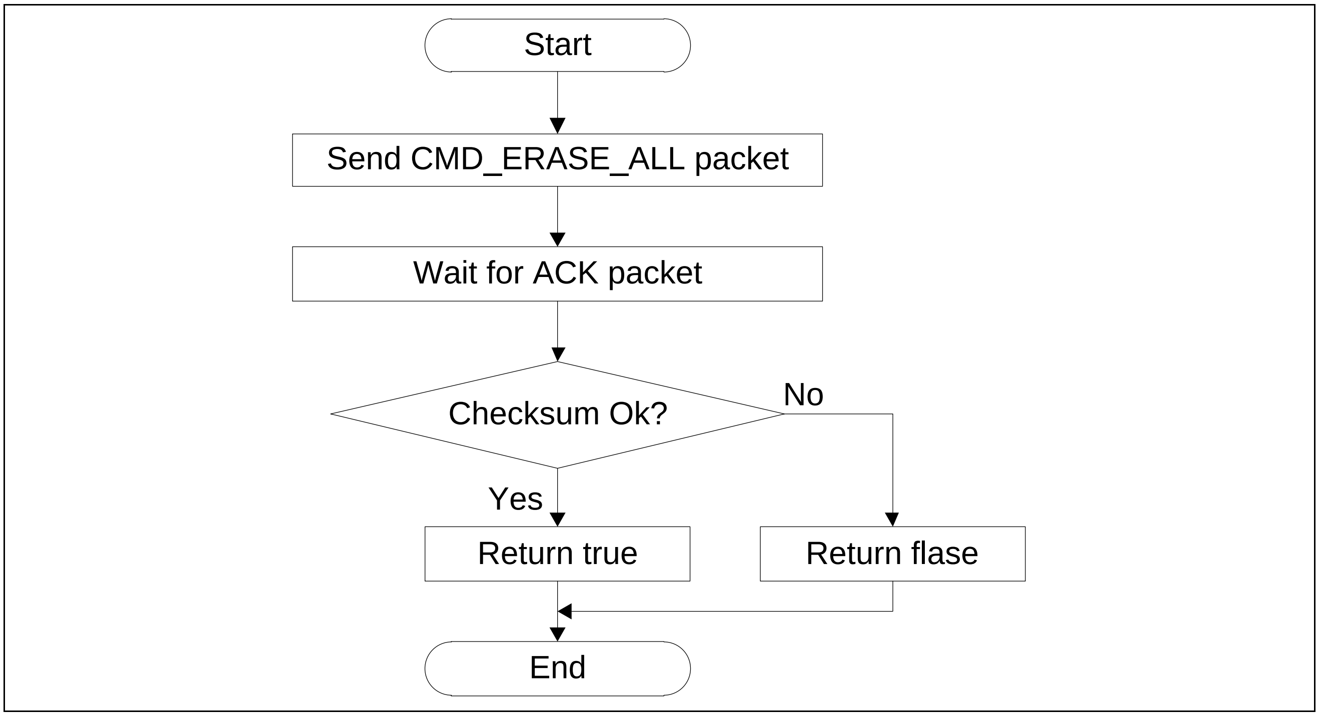
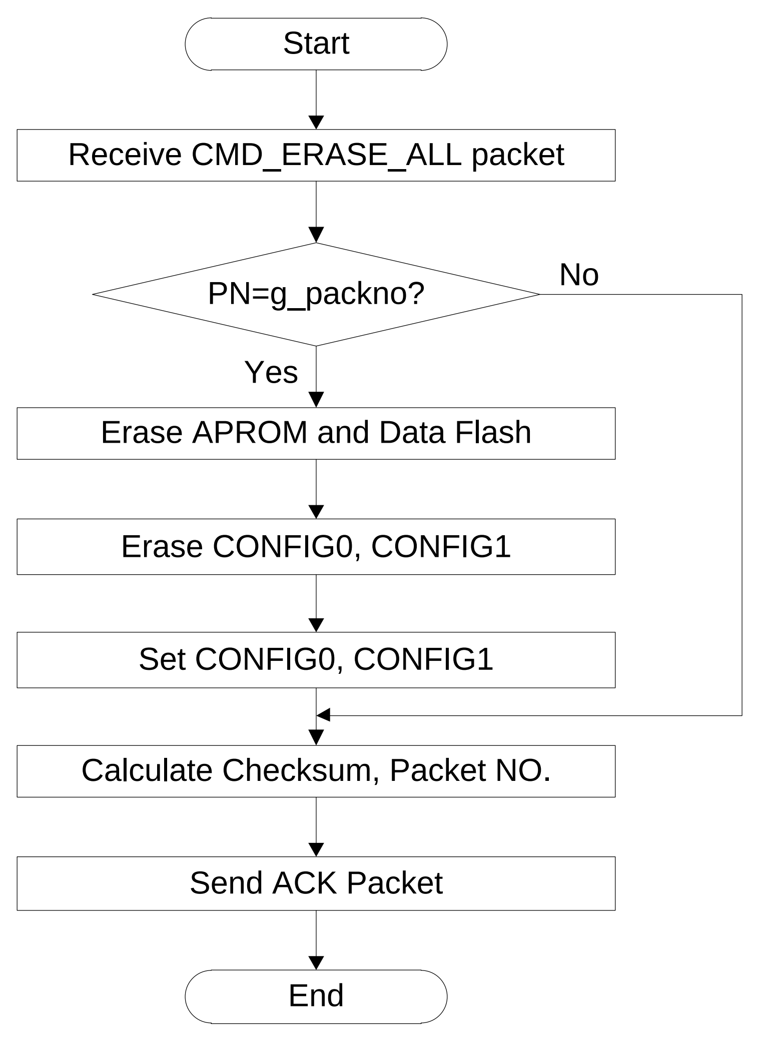


Figure 3-8 The Flow of Host Erase APROM


Start
Receive CMD_ERASE_ALL packet
PN=g_packno?
No
Yes
Erase APROM and Data Flash
Erase CONFIG0, CONFIG1
Set CONFIG0, CONFIG1
Calculate Checksum, Packet NO.
Send ACK Packet
End

Note: PN is packet number received from host；g_packno is device local packet number.

Figure 3-9 The Flow of Device Erase APROM

3.4.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**

0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000A3

3.4.2 Device Replay Packet Format

63	**62**	**61**	**60**	**59**	**58**	**57**	**56**
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**

0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				Checksum			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Checksum**	Sum up byte 0 to 11 of the receive package.

3.5 CMD_SYNC_PACKNO Command Format

This command is used to sync up the ISP packet number with MCU. Before sending any command, the host, ISP tool, needs to send the command to sync up the packet number (PN) with the device, MCU. When the device receives the command packet, it will check whether the received PN and RN are equal or not. If they are equal, the device calculates the checksum of the whole packet received and sets the received PN increased by one as the PN of ACK packet. Then transmits ACK packet to the host. If the PN is not equal to the RN, the device will discard this package and send ACK package with the PN as zero.

After ACK packet is received, host will check whether the checksum is right or not. If the checksum is right, it indicates that the connection with the device has established successfully. After this operation is finished, users can send command packet to start the corresponding operation. Otherwise, if checksum is wrong, it indicates that the connection with the device is failed, and host will not send any other command packet.

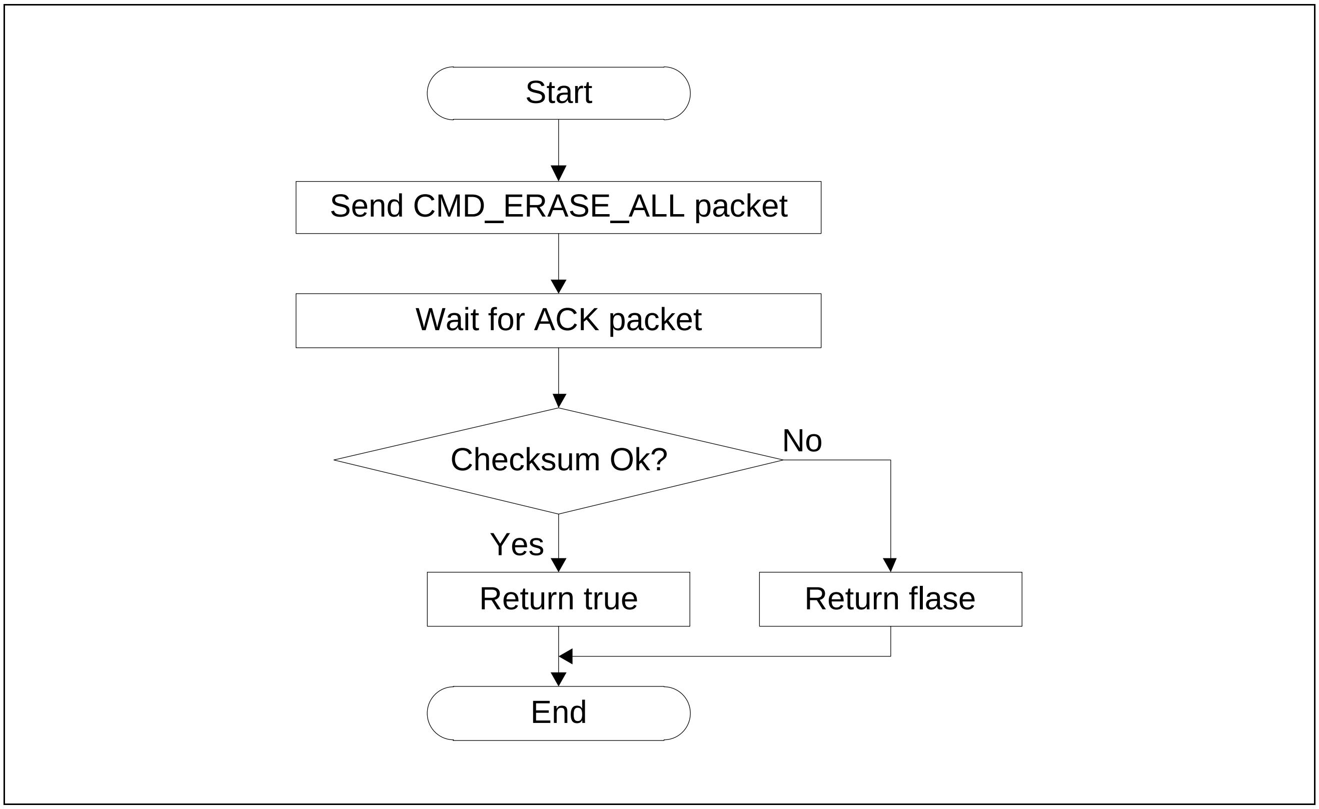


Figure 3-10 The Flow of Host Sync up Packet Number

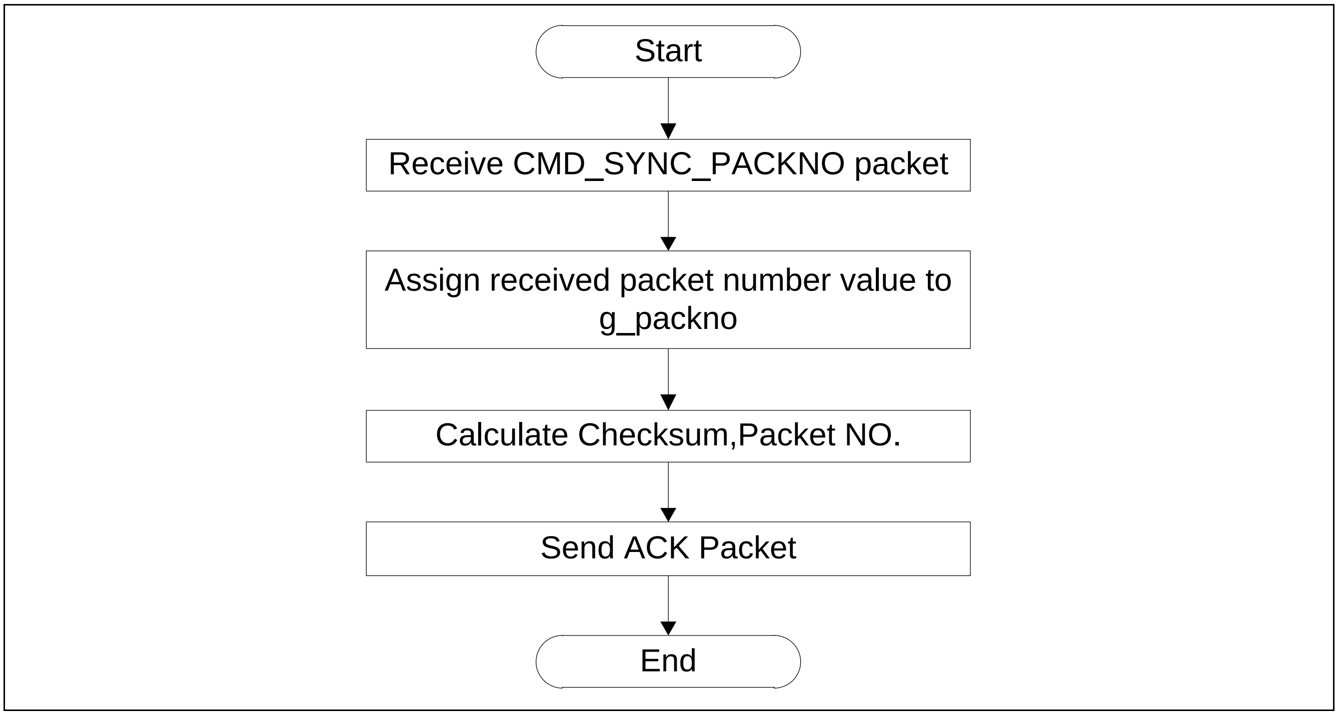

Start
Receive CMD_SYNC_PACKNO packet
Assign received packet number value to g_packno
Calculate Checksum,Packet NO.
Send ACK Packet
End

Figure 3-11 The Flow of Device Sync up Packet Number

3.5.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0				RN			

7	6	5	4	3	2	1	0
Packet Number				CMD			

Byte	Name	Description
[63:12]	**Reserved**	Reserved
[11:8]	**RN**	It is equal to Packet Number.
[7:4]	**Packet Number**	It is a sequential number.
[3:0]	**CMD**	Command code is 0x000000A4

3.5.2 Device Replay Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				Checksum			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Checksum**	Sum up byte 0 to 11 of the receive package.

3.6 CMD_GET_FWVER Command

This command is used to get version information of ISP firmware. When the device receives the command packet, the device gets the information of version and then fills the information into ACK packet. After this, the device calculates the checksum of the whole packet received, then transmits ACK packet to the host.

After ACK packet is received, host will check whether the checksum is right or not. If the checksum is right, it indicates that the data transfer is successful, then users can retrieve the version of ISP firmware from the received packet. After the operation is finished, users can send another command packet to start another operation.

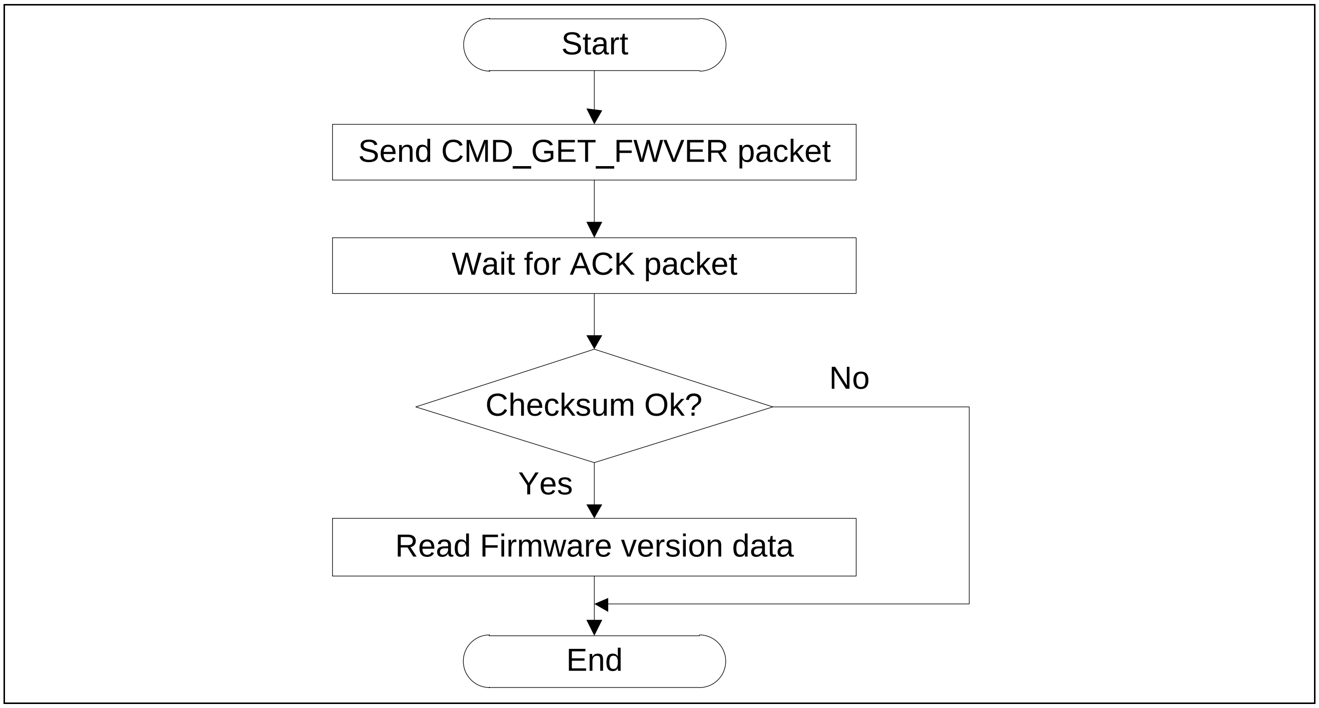
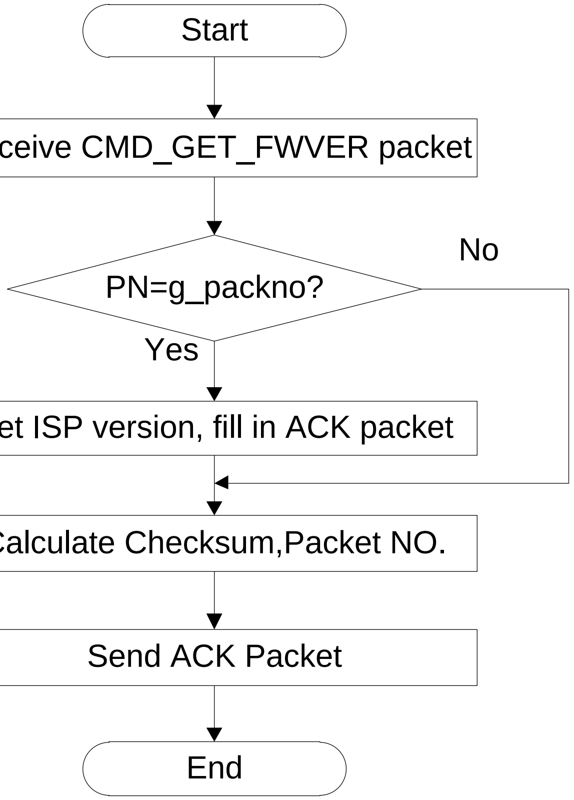


Figure 3-12 The Flow of Get ISP Firmware Version Command on the Host Side


Start
Receive CMD_GET_FWVER packet
PN=g_packno?
No
Yes
Set ISP version, fill in ACK packet
Calculate Checksum,Packet NO.
Send ACK Packet
End

Note: PN is packet number received from host；g_packno is device local packet number.

Figure 3-13 The Flow of Get ISP Firmware Version Command on the Device Side

3.6.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**

<table>
<tr><td colspan="8">0</td></tr>
<tr><td><b>15</b></td><td><b>14</b></td><td><b>13</b></td><td><b>12</b></td><td><b>11</b></td><td><b>10</b></td><td><b>9</b></td><td><b>8</b></td></tr>
<tr><td colspan="8">0</td></tr>
<tr><td><b>7</b></td><td><b>6</b></td><td><b>5</b></td><td><b>4</b></td><td><b>3</b></td><td><b>2</b></td><td><b>1</b></td><td><b>0</b></td></tr>
<tr><td colspan="4">Packet Number</td><td colspan="4">CMD</td></tr>
</table>

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000A6

3.6.1 Device Replay Packet Format

<table>
<tr><td><b>63</b></td><td><b>62</b></td><td><b>61</b></td><td><b>60</b></td><td><b>59</b></td><td><b>58</b></td><td><b>57</b></td><td><b>56</b></td></tr>
<tr><td colspan="8">0</td></tr>
<tr><td><b>55</b></td><td><b>54</b></td><td><b>53</b></td><td><b>52</b></td><td><b>51</b></td><td><b>50</b></td><td><b>49</b></td><td><b>48</b></td></tr>
<tr><td colspan="8">0</td></tr>
<tr><td><b>47</b></td><td><b>46</b></td><td><b>45</b></td><td><b>44</b></td><td><b>43</b></td><td><b>42</b></td><td><b>41</b></td><td><b>40</b></td></tr>
<tr><td colspan="8">0</td></tr>
<tr><td><b>39</b></td><td><b>38</b></td><td><b>37</b></td><td><b>36</b></td><td><b>35</b></td><td><b>34</b></td><td><b>33</b></td><td><b>32</b></td></tr>
<tr><td colspan="8">0</td></tr>
<tr><td><b>31</b></td><td><b>30</b></td><td><b>29</b></td><td><b>28</b></td><td><b>27</b></td><td><b>26</b></td><td><b>25</b></td><td><b>24</b></td></tr>
<tr><td colspan="8">0</td></tr>
<tr><td><b>23</b></td><td><b>22</b></td><td><b>21</b></td><td><b>20</b></td><td><b>19</b></td><td><b>18</b></td><td><b>17</b></td><td><b>16</b></td></tr>
<tr><td colspan="8">0</td></tr>
<tr><td><b>15</b></td><td><b>14</b></td><td><b>13</b></td><td><b>12</b></td><td><b>11</b></td><td><b>10</b></td><td><b>9</b></td><td><b>8</b></td></tr>
<tr><td colspan="4">0</td><td colspan="4">Version</td></tr>
<tr><td><b>7</b></td><td><b>6</b></td><td><b>5</b></td><td><b>4</b></td><td><b>3</b></td><td><b>2</b></td><td><b>1</b></td><td><b>0</b></td></tr>
<tr><td colspan="4">Packet Number</td><td colspan="4">Checksum</td></tr>
</table>

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[11:8]	**Version**	Firmware version information.
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Checksum**	Sum up byte 0 to 11 of the receive package.

3.7 CMD_GET_ DEVICEID Command Format

This command is used to get the device ID. ISP programming tool needs this ID to inquire the size of APROM and then informs it to the device . When the device receives the command packet, it gets device ID and then fills it in ACK packet. After this, the device calculates the checksum of the whole packet received, then transmits ACK packet to the host.

After ACK packet is received, host will check whether the checksum is right or not. If the checksum is right, it indicates that the data transfer is successful, then users can read the device ID from received packet. After the operation is finished, users can send another command packet to start another operation.

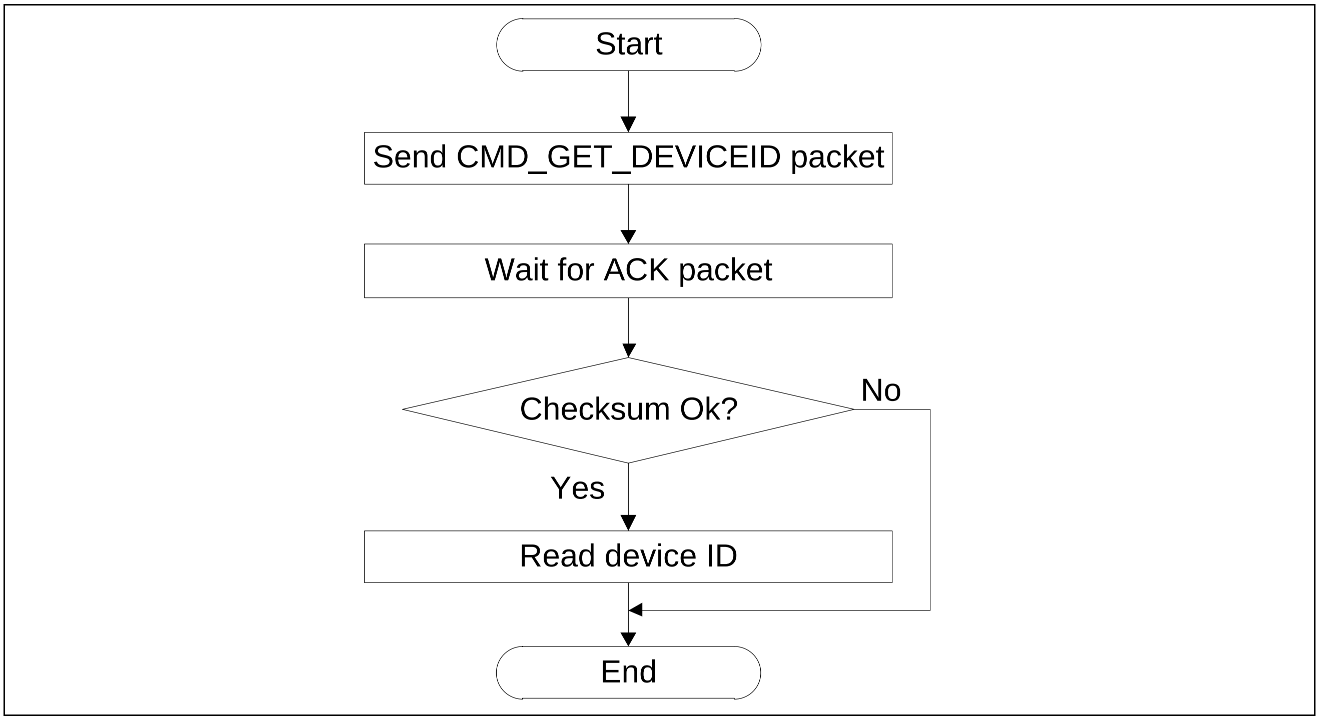
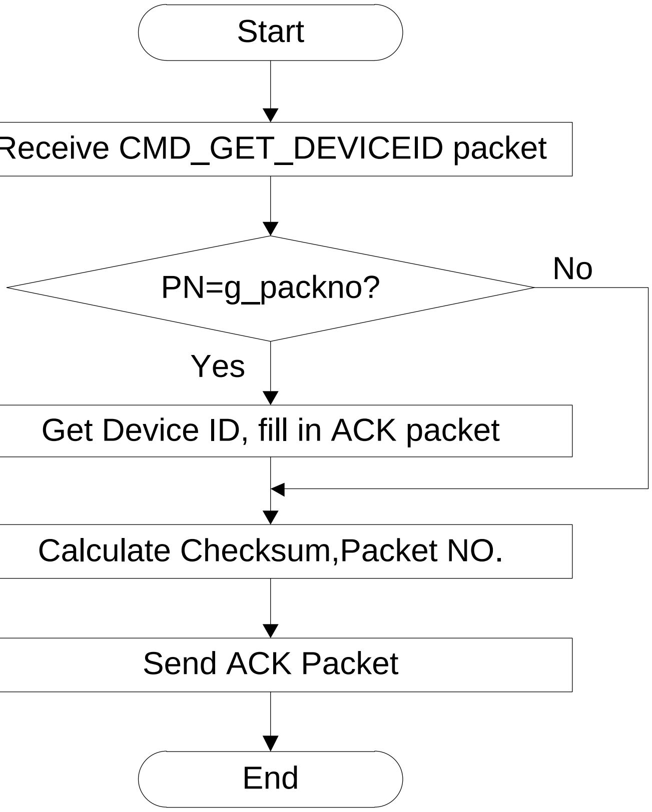


Figure 3-14 The Flow of Host Get Device ID


Start
Receive CMD_GET_DEVICEID packet
PN=g_packno?
No
Yes
Get Device ID, fill in ACK packet
Calculate Checksum,Packet NO.
Send ACK Packet
End

Note: PN is packet number received from host； g_packno is device local packet number.

Figure 3-15 The Flow of Device Send Device ID

3.7.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**

0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000B1

3.7.2 Device Replay Packet Format

63	**62**	**61**	**60**	**59**	**58**	**57**	**56**
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0				Device ID			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				Checksum			

Byte	Name	Description
[63:12]	**Reserved**	Reserved
[11:8]	**Device ID**	PID
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**Checksum**	Sum up byte 0 to 11 of the receive package.

3.8 CMD_RUN_APROM Command Format

This command is used to set the system to boot from APROM. If users want to run the application in APROM, the device can be set to execute this command in boot code. After this, the device will be rebooted from APROM.

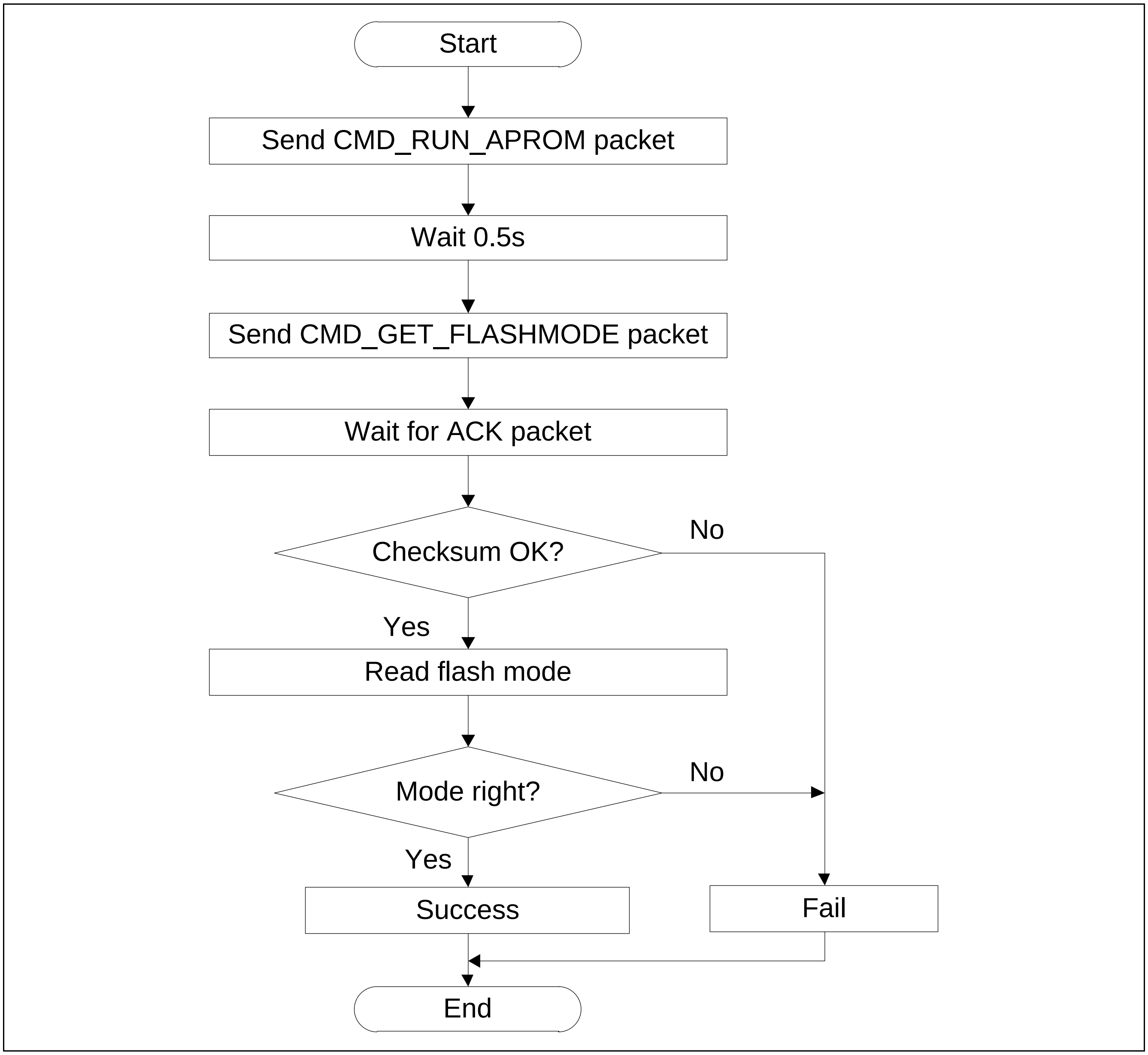
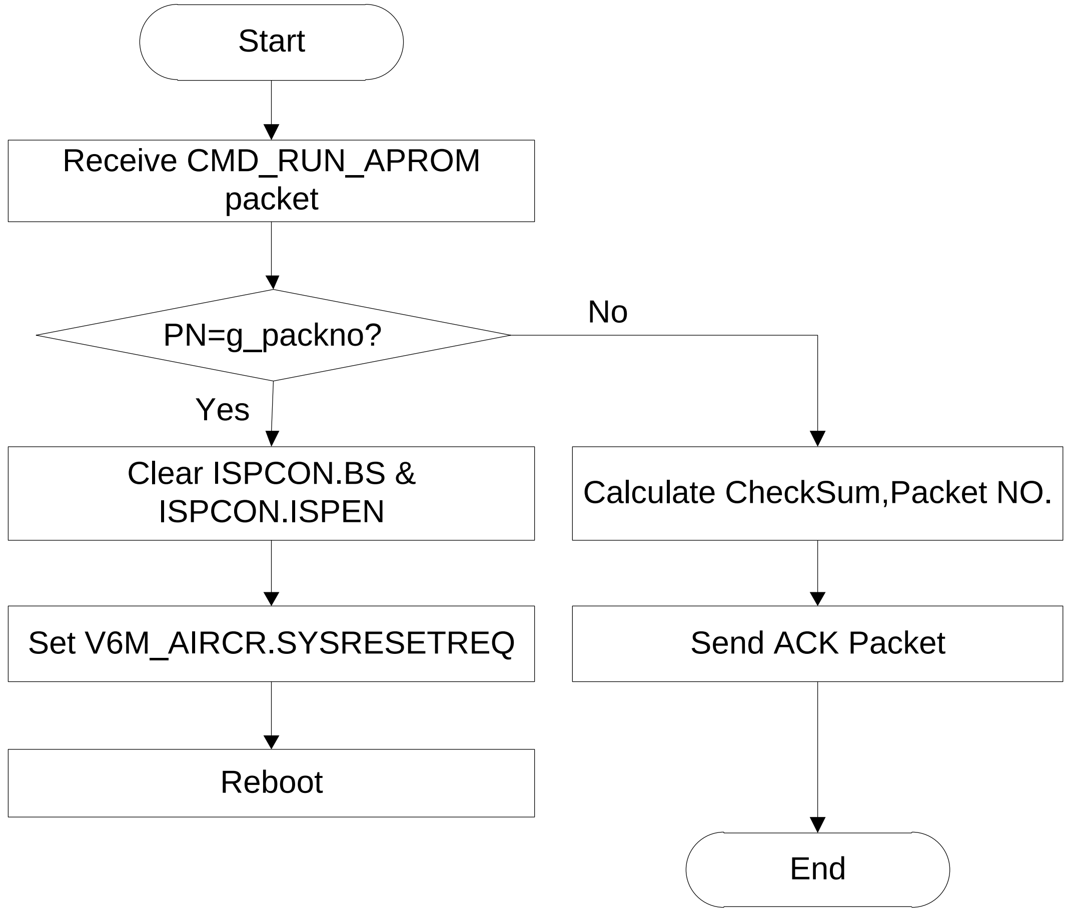


Figure 3-16 The Flow of Host Update APROM


Start
Receive CMD_RUN_APROM packet
PN=g_packno?
No
Yes
Clear ISPCON.BS & ISPCON.ISPEN
Calculate CheckSum,Packet NO.
Set V6M_AIRCR.SYSRESETREQ
Send ACK Packet
Reboot
End

Note: PN is packet number received from host；g_packno is device local packet number.

Figure 3-17 The Flow of Device Update APROM

3.8.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**

0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000AB

3.8.2 Device Replay Packet Format

Device will wait about 0.5 s, then the system will restart from APROM.

3.9 CMD_RUN_LDROM Command Format

This command is used to set the system to boot from LDROM. If users want to run the boot code in LDROM, the device can be set to execute this command in application code. After this, the device will be rebooted from LDROM.

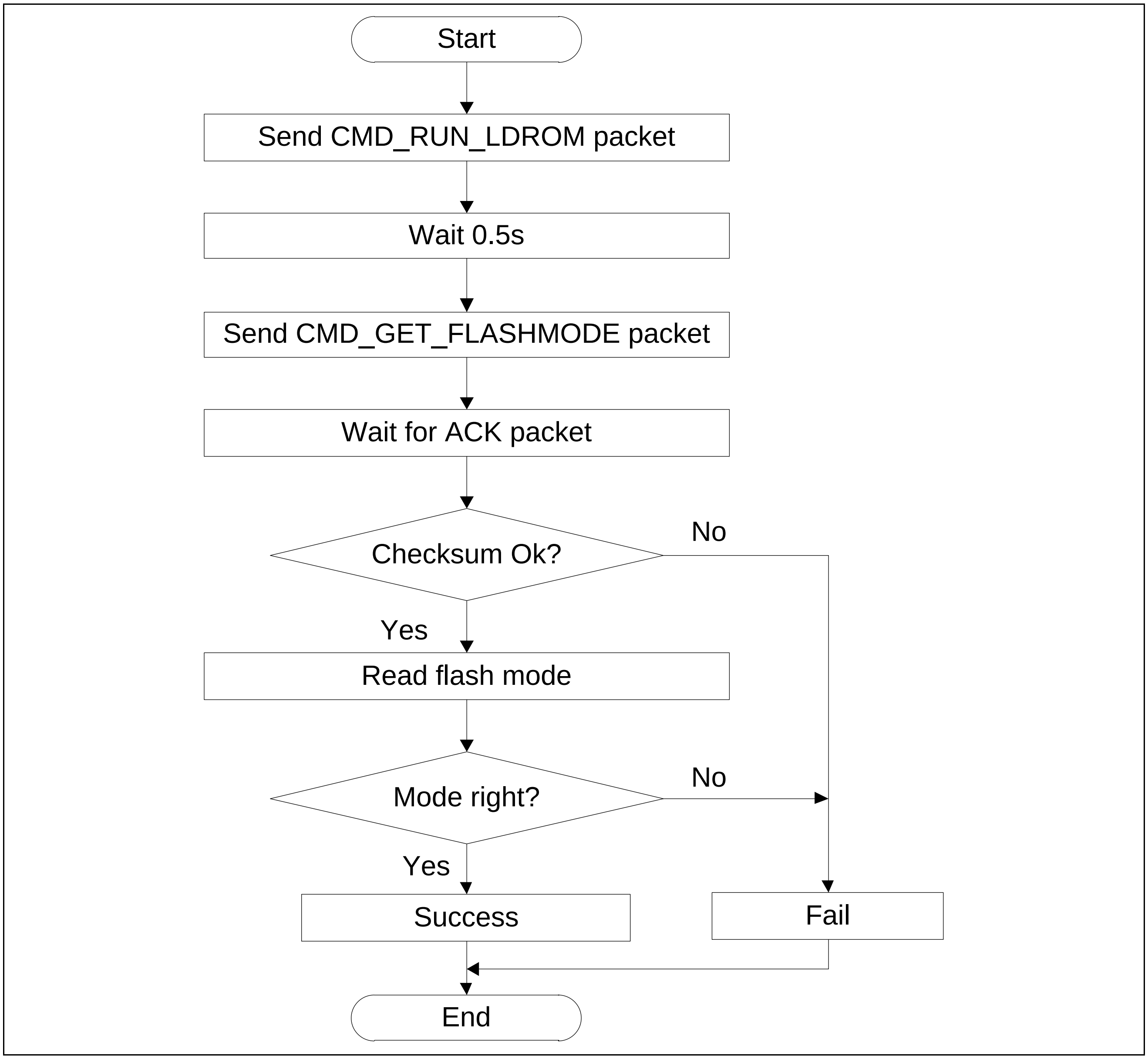
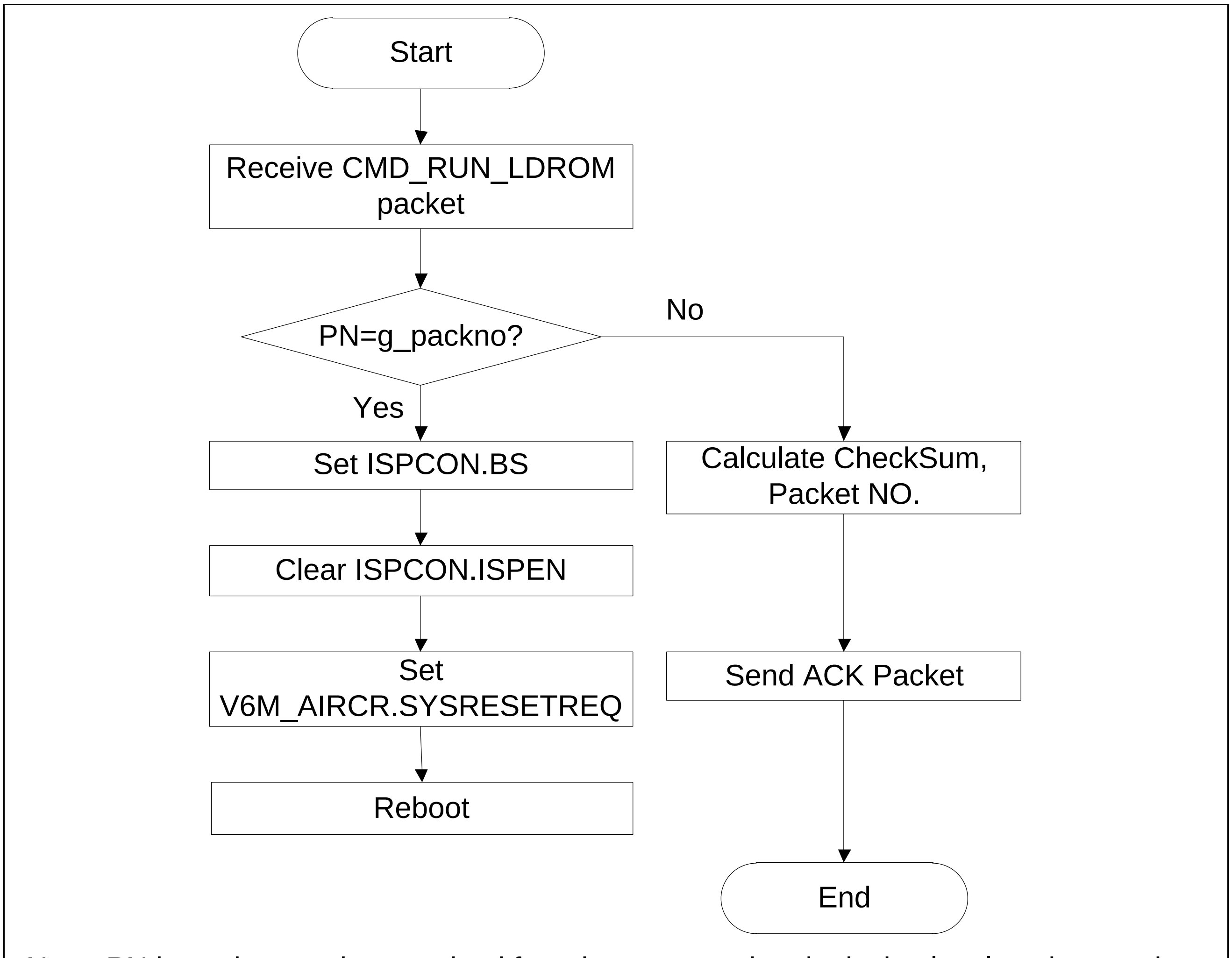


Figure 3-18 The Flow of Host Run LDROM


Start
Receive CMD_RUN_LDROM packet
PN=g_packno?
No
Yes
Set ISPCON.BS
Clear ISPCON.ISPEN
Set V6M_AIRCR.SYSRESETREQ
Reboot
Calculate CheckSum, Packet NO.
Send ACK Packet
End

Note: PN is packet number received from host; g_packno is device local packet number.

Figure 3-19 The Flow of Device Run LDROM

3.9.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							

31	30	29	28	27	26	25	24
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000AC

3.9.2 Device Replay Packet Format

Device will wait about 0.5 s, then the system will restart from LDROM.

3.10 CMD_CONNECT Command Format

The command is used to check whether the device is currently running boot code or not. If the device is running boot code now, the device will reply with 64-byte ACK packet. When device receives the command packet, it will restore g_packno to 1 and return ACK packet to the host.

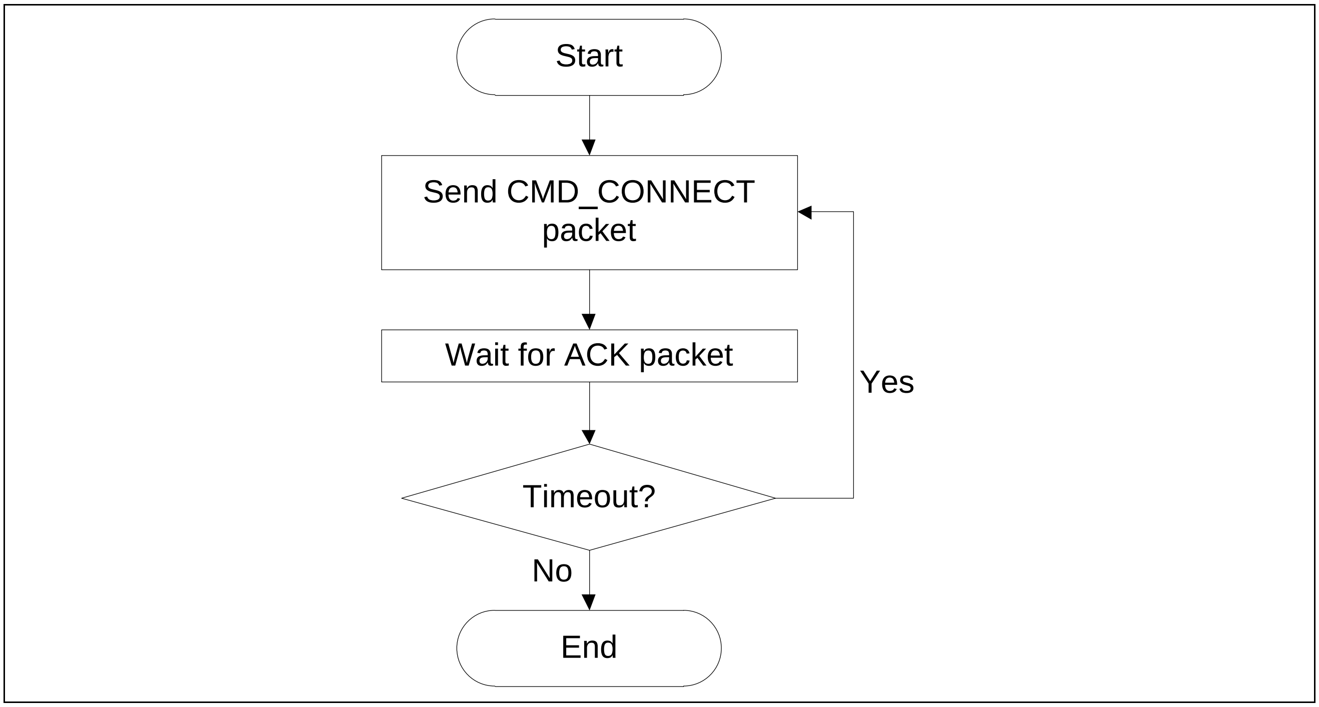
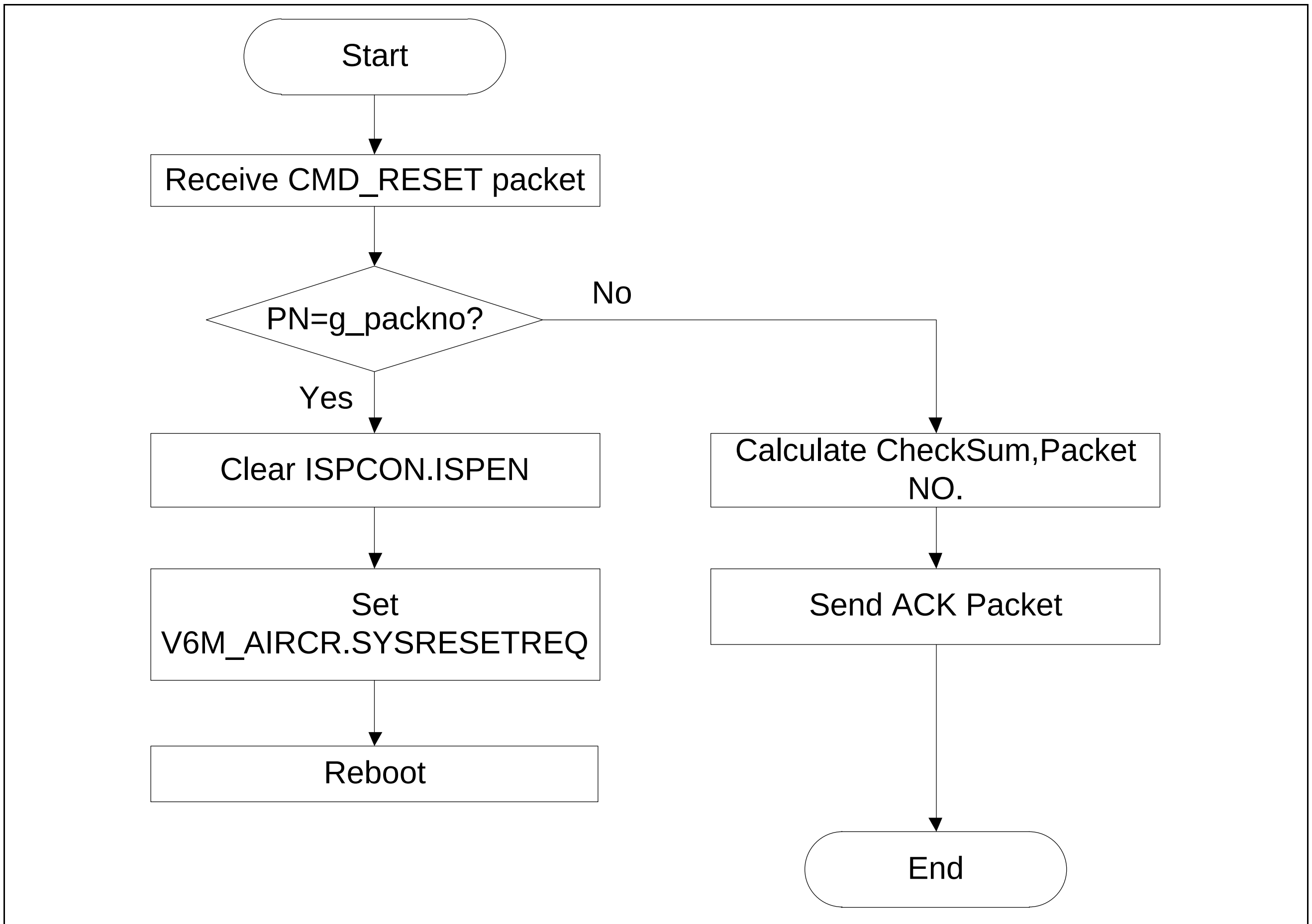


Figure 3-20 The Flow of Host Connect with Device



Note: PN is packet number received from host； g_packno is device local packet number.

Figure 3-21 The Flow of Device Connect with Host

3.10.1 Host Request Packet Format

63	62	61	60	59	58	57	56
0							
55	54	53	52	51	50	49	48
0							
47	46	45	44	43	42	41	40
0							
39	38	37	36	35	34	33	32
0							
31	30	29	28	27	26	25	24

0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Packet Number				CMD			

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increasing by 1 per packet.
[3:0]	**CMD**	Command code is 0x000000AE

3.10.2 Device Replay Packet Format

63	**62**	**61**	**60**	**59**	**58**	**57**	**56**
0							
55	**54**	**53**	**52**	**51**	**50**	**49**	**48**
0							
47	**46**	**45**	**44**	**43**	**42**	**41**	**40**
0							
39	**38**	**37**	**36**	**35**	**34**	**33**	**32**
0							
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Packet Number	Checksum

Byte	Name	Description
[63:8]	**Reserved**	Reserved
[7:4]	**Packet Number**	It is a sequential number increased by 1.
[3:0]	**Checksum**	Sum up byte 0 to 11 of the receive package.

4 Quick Start - ISP Programming through Nu-Link2-Pro

This chapter introduces how to connect the Nu-Link2-Pro to a computer, and how to set the Nu-Link2-Pro as the third-party tool.

Note: For more details, please refer to the Nu-Link2-Pro User Manual.

4.1 Tool Chain Supporting

This section briefly describes required software settings for connecting to the Nu-Link2-Pro. For detailed software operation, please refer to the related user manuals.

4.1.1 Keil C51

Please refer to Keil C51, and make sure the Nu-Link_Keil_Driver for Keil C51 has been downloaded and installed such that Keil C51 can recognize the Nu-Link2-Pro.

4.1.2 IAR EW8051

Please refer to IAR EW8051, and make sure that Nu-Link_IAR_Driver for IAR EW8051 has been downloaded and installed before setting the Nu-Link2-Pro such that IAR EW8051 can recognize the Nu-Link2-Pro.

4.2 ICP Programming

4.2.1 ICP Programming Connection

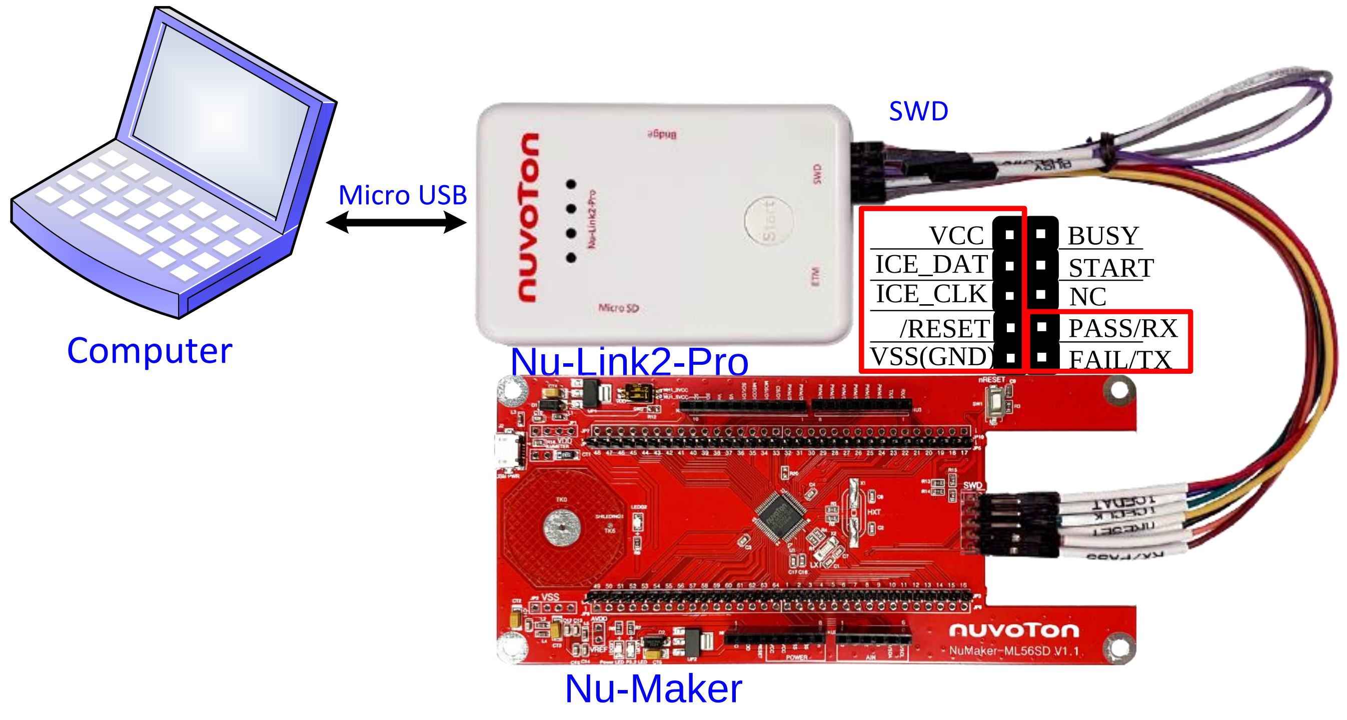
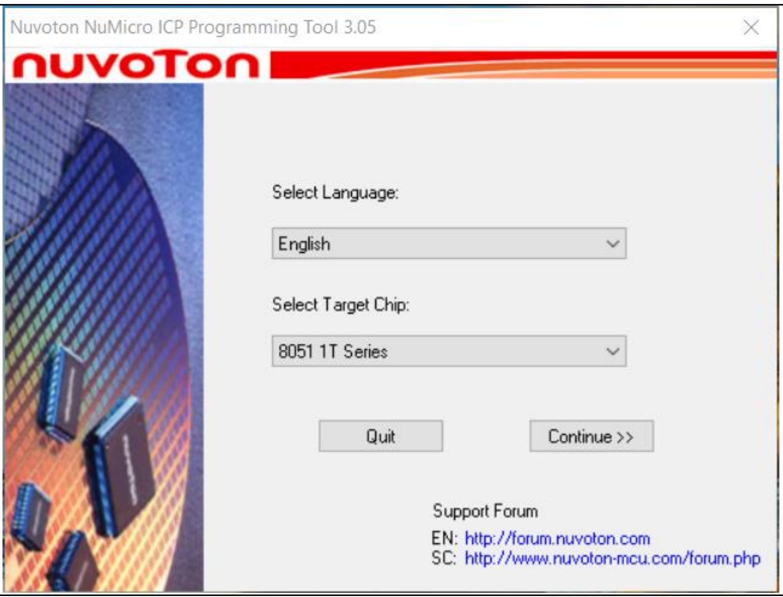


Figure 4-1 ICP Programming Connection

4.2.2 ICP Programming Tool

(1) Download and install [Nuvoton NuMicro® ICP Programming Tool]().

(2) Open the ICP Tool and select the **UI language** and **target chip**, and then click **Continue**, as shown in Figure 4-2.



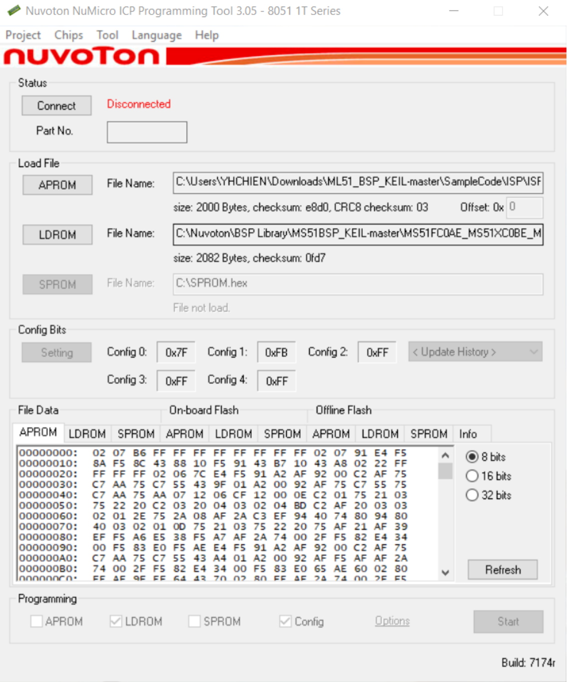
Select Language: English

Select Target Chip: 8051 1T Series

Support Forum
EN: http://forum.nuvoton.com
SC: http://www.nuvoton-mcu.com/forum.php

Figure 4-2 Startup Screen of ICP Tool

(3) In the ICP Tool main window, the connection status is shown as "Disconnected" since the ICP tool has not been connected with the Nu-Link2-Pro, as shown in Figure 4-3.



Status: Disconnected

Part No.:

APROM File Name: C:\Users\YHCHIEN\Downloads\ML51_BSP_KEIL-master\SampleCode\ISP\ISF

size: 2000 Bytes, checksum: e8d0, CRC8 checksum: 03 Offset: 0x 0

LDROM File Name: C:\Nuvoton\BSP Library\MS51BSP_KEIL-master\MS51FC0AE_MS51XC0BE_M

size: 2082 Bytes, checksum: 0fd7

SPROM File Name: C:\SPROM.hex

File not load.

Config 0: 0x7F Config 1: 0xFB Config 2: 0xFF

Config 3: 0xFF Config 4: 0xFF

```
00000000:  02 07 B6 FF FF FF FF FF FF FF FF 02 07 91 E4 F5
00000010:  8A F5 8C 43 88 10 F5 91 43 B7 10 43 A8 02 22 FF
00000020:  FF FF FF 02 06 7C E4 F5 91 A2 AF 92 00 C2 AF 75
00000030:  C7 AA 75 C7 55 43 9F 01 A2 00 92 AF 75 C7 55 75
00000040:  C7 AA 75 AA 07 12 06 CF 12 00 0E C2 01 75 21 03
00000050:  75 22 20 C2 03 20 04 03 02 04 BD C2 AF 20 03 03
00000060:  02 01 2E 75 2A 08 AF 2A C3 EF 94 40 74 80 94 80
00000070:  40 03 02 01 0D 75 21 03 75 22 20 75 AF 21 AF 39
00000080:  EF F5 A6 E5 38 F5 A7 AF 2A 74 00 2F F5 82 E4 34
00000090:  00 F5 83 E0 F5 AE E4 F5 91 A2 AF 92 00 C2 AF 75
000000A0:  C7 AA 75 C7 55 43 A4 01 A2 00 92 AF F5 AF AF 2A
000000B0:  74 00 2F F5 82 E4 34 00 F5 83 E0 65 AE 60 02 80
```

● 8 bits
○ 16 bits
○ 32 bits

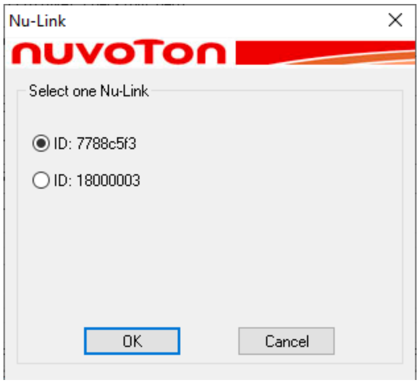
☐ APROM ☑ LDROM ☐ SPROM ☑ Config

Build: 7174r

Figure 4-3 ICP Tool Main Window

(4) And then click the **Connect** button. Go to (5) if more are connected with the host computer than one Nu-Link Debugger and Programmer, such as Nu-Link2-Pro. Go to (6) if only one Nu-Link2-Pro is connected with the host computer.

(5) If multiple Nu-Link Debugger and Programmers have been connected with the host computer, a message appears and asks users to make a selection from one of these Nu-Link Debugger and Programmers. Clicking **OK** will connect the selected adapter with the host, as shown in Figure 4-4. When a Nu-Link Debugger and Programmer is selected and connects to the host successfully, the Status LED starts blinking.



Select one Nu-Link

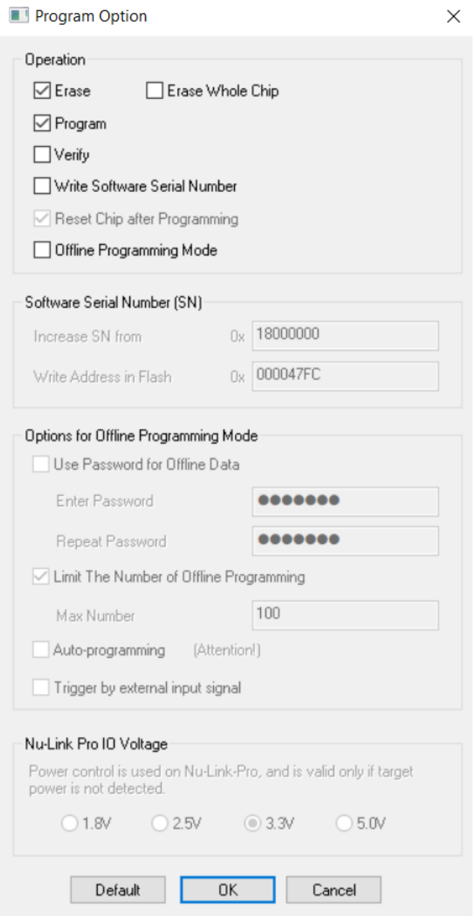
● ID: 7788c5f3

○ ID: 18000003

Figure 4-4 Select One Nu-Link Debugger and Programmer

(6) Click **Option** in the **Programming** section of the ICP Tool Main Window to open the **Program Option** form, as shown in Figure 4-5**.**

(7) In the **Nu-Link Pro IO Voltage** section, select the power voltage of the SWD port for the target chip, and then click **OK**. To use the offline programming function, the **Offline Programming Mode** option in **Operation** section should be selected, as shown in Figure 4-5.



Program Option

Operation

☑ Erase ☐ Erase Whole Chip
☑ Program
☐ Verify
☐ Write Software Serial Number
☑ Reset Chip after Programming
☐ Offline Programming Mode

Software Serial Number (SN)

Increase SN from 0x 18000000
Write Address in Flash 0x 000047FC

Options for Offline Programming Mode

☐ Use Password for Offline Data
Enter Password ●●●●●●●
Repeat Password ●●●●●●●
☑ Limit The Number of Offline Programming
Max Number 100
☐ Auto-programming (Attention!)
☐ Trigger by external input signal

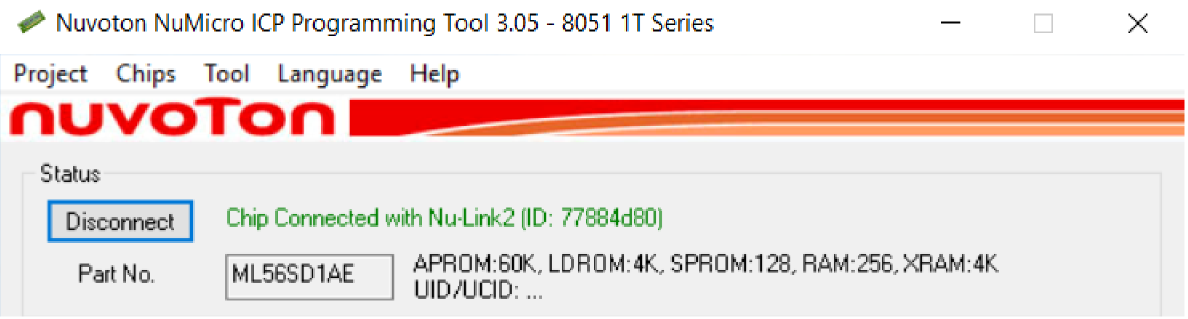
Nu-Link Pro IO Voltage

Power control is used on Nu-Link-Pro, and is valid only if target power is not detected.

○ 1.8V ○ 2.5V ● 3.3V ○ 5.0V

Figure 4-5 ICP Tool Program Option Form

(8) After the **Connect** button is clicked, the ICP Tool will be connected with the Nu-Link2-Pro, and an SWD port will be detected. Figure 4-6 shows that the ICP Tool has been connected with the Nu-Link2-Pro and a target chip is detected. Meanwhile, users can start programming the target chip.



Nuvoton NuMicro ICP Programming Tool 3.05 - 8051 1T Series

Chip Connected with Nu-Link2 (ID: 77884d80)

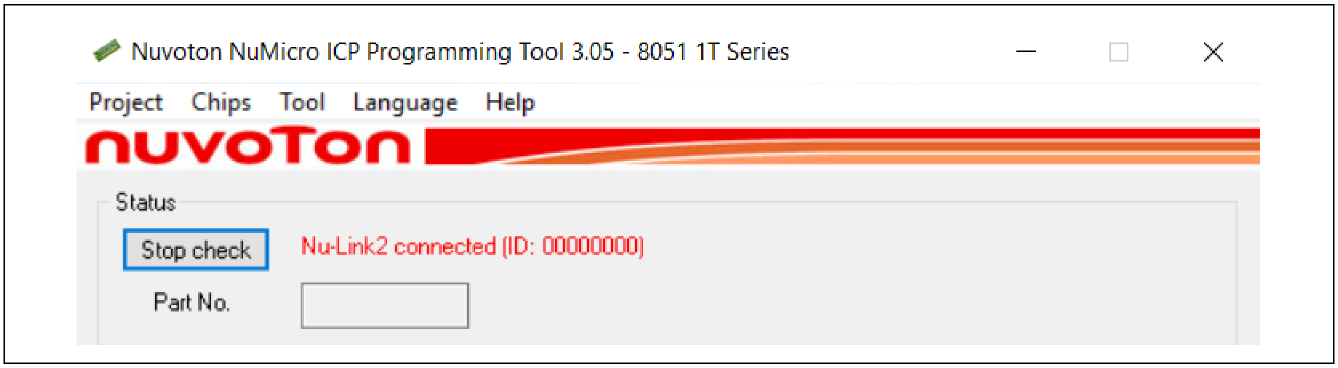
Part No. ML56SD1AE

APROM:60K, LDROM:4K, SPROM:128, RAM:256, XRAM:4K
UID/UCID: ...

Figure 4-6 Nu-Link2-Pro Connected with a Target Chip Detected

(9) Figure 4-7 shows that the ICP Tool has been connected with the Nu-Link2-Pro yet no target

chip is detected. The ICP tool will continue detecting the target chip until the **Stop Check** button is clicked. During this period, users cannot program any chips, but users can save the offline programming information in the Nu-Link2-Pro.



Nu-Link2 connected (ID: 00000000)

Figure 4-7 Nu-Link2-Pro Connected with No Target Chip Detected

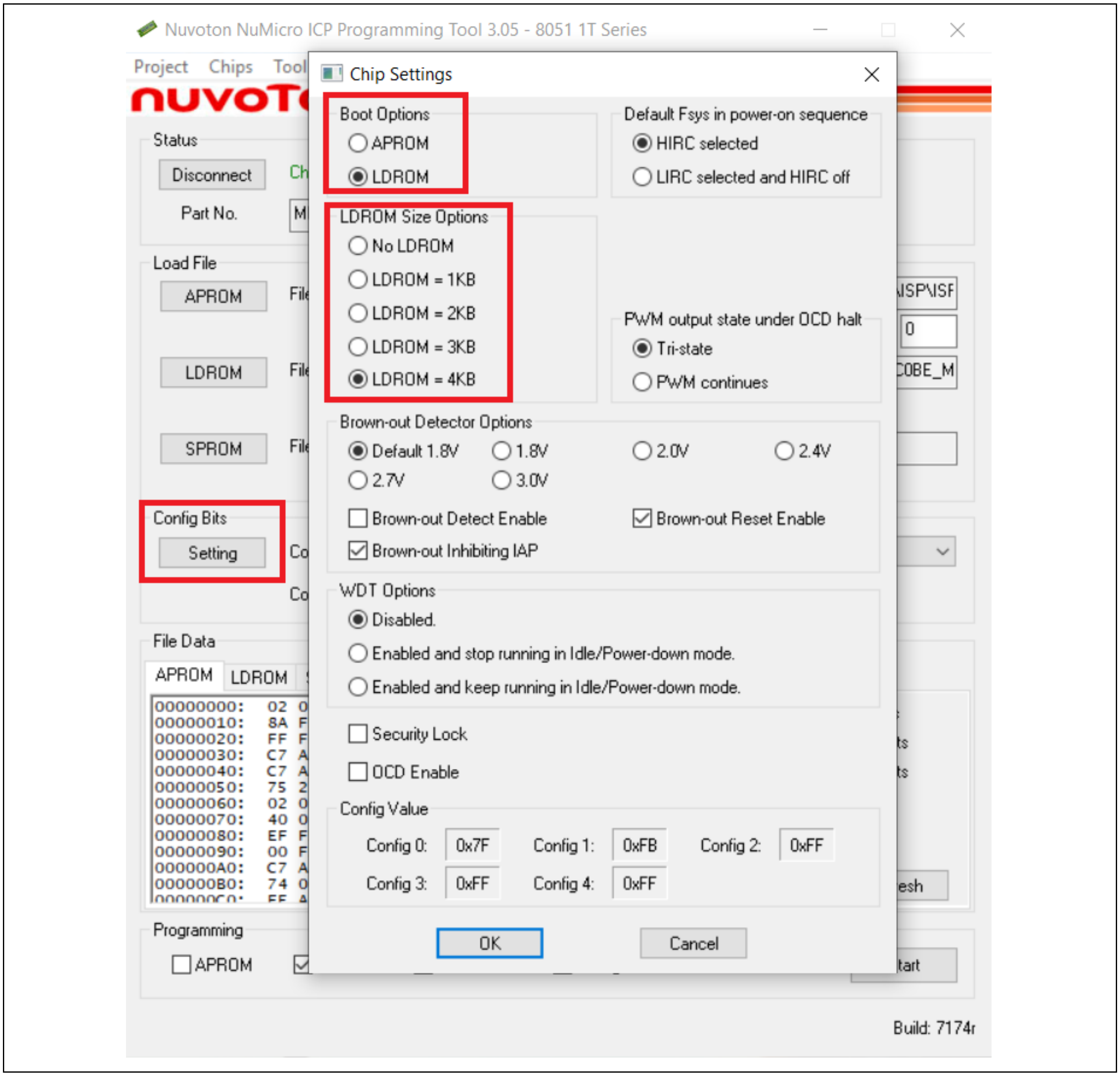
(10) Click the **Disconnect** button if programming is not required, as shown in Figure 4-6. Or click the **Stop Check** button to disconnect the ICP Tool from the Nu-Link2-Pro, as shown in Figure 4-7. After the disconnection, the Nu-Link2-Pro can resume a connection with another tool.

(11) Click the **Setting** button in the **Config Bits** section to open the **Chip Settings** form, and select **LDROM** in **Boot Options** and set LDROM size in **LDROM Size Options** according to your need, as shown in Figure 4-8.

(12) As shown in Figure 4-9, in the Programming File Selection Window, select the required ".bin" file to program LDROM, and tick the radio button next to the memory location aimed to program, which is LDROM in this case.

(13) Click the **Start** button in the Programming File Selection Window to start programming.

(14) As shown in Figure 4-10, click the **OK** button in the Programming Completion Window to complete the operation.



Chip Settings

Boot Options
○ APROM
● LDROM

Default Fsys in power-on sequence
● HIRC selected
○ LIRC selected and HIRC off

LDROM Size Options
○ No LDROM
○ LDROM = 1KB
○ LDROM = 2KB
○ LDROM = 3KB
● LDROM = 4KB

PWM output state under OCD halt
● Tri-state
○ PWM continues

Brown-out Detector Options
● Default 1.8V ○ 1.8V ○ 2.0V ○ 2.4V
○ 2.7V ○ 3.0V
☐ Brown-out Detect Enable ☑ Brown-out Reset Enable
☑ Brown-out Inhibiting IAP

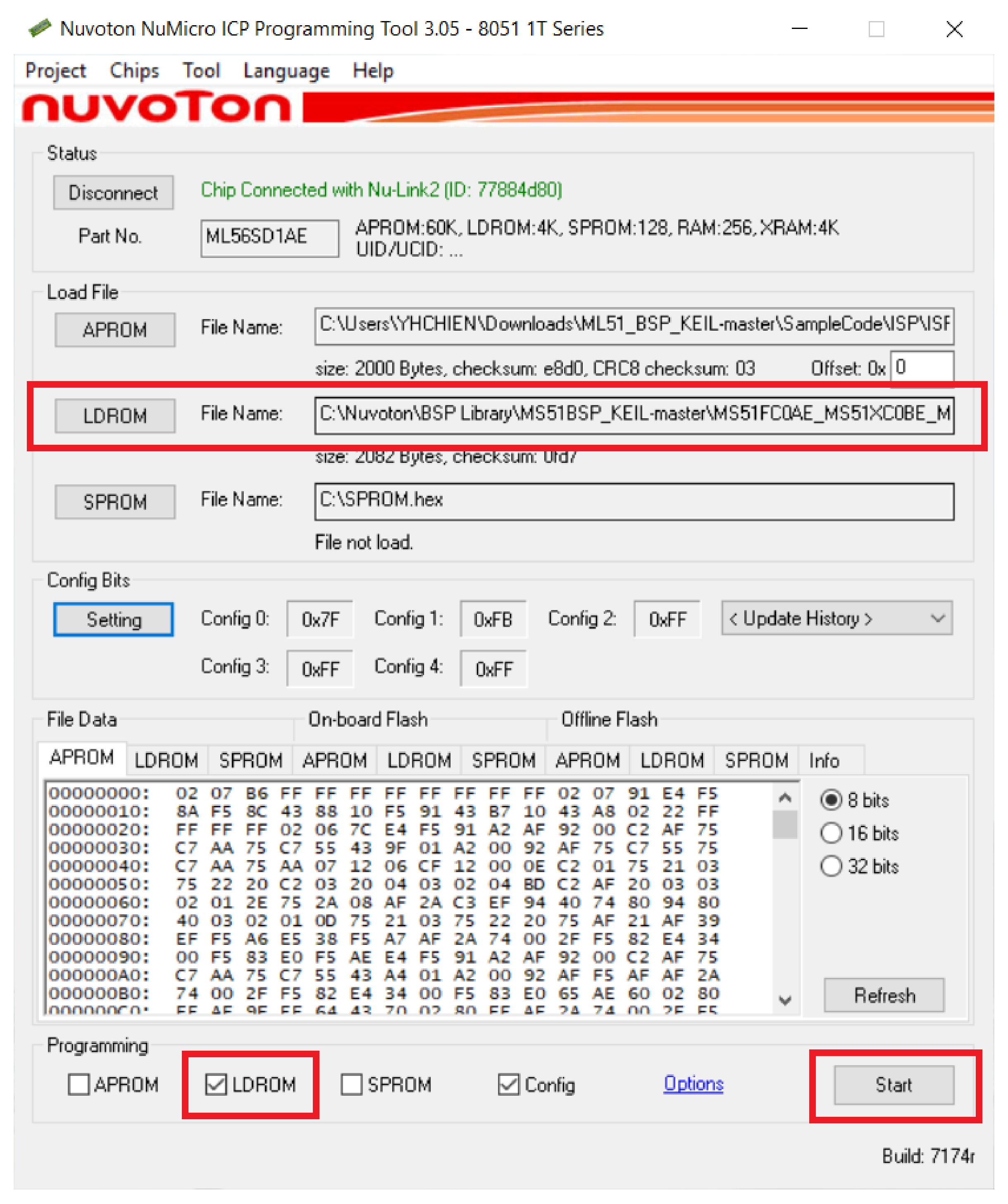
WDT Options
● Disabled.
○ Enabled and stop running in Idle/Power-down mode.
○ Enabled and keep running in Idle/Power-down mode.

☐ Security Lock
☐ OCD Enable

Config Value
Config 0: 0x7F Config 1: 0xFB Config 2: 0xFF
Config 3: 0xFF Config 4: 0xFF

OK Cancel

Figure 4-8 Config Bits Setting



Chip Connected with Nu-Link2 (ID: 77884d80)

Part No. ML56SD1AE

APROM:60K, LDROM:4K, SPROM:128, RAM:256, XRAM:4K
UID/UCID: ...

File Name: C:\Users\YHCHIEN\Downloads\ML51_BSP_KEIL-master\SampleCode\ISP\ISF

size: 2000 Bytes, checksum: e8d0, CRC8 checksum: 03 Offset: 0x 0

File Name: C:\Nuvoton\BSP Library\MS51BSP_KEIL-master\MS51FC0AE_MS51XC0BE_M

size: 2082 Bytes, checksum: 0fd7

File Name: C:\SPROM.hex

File not load.

Config 0: 0x7F Config 1: 0xFB Config 2: 0xFF < Update History >

Config 3: 0xFF Config 4: 0xFF

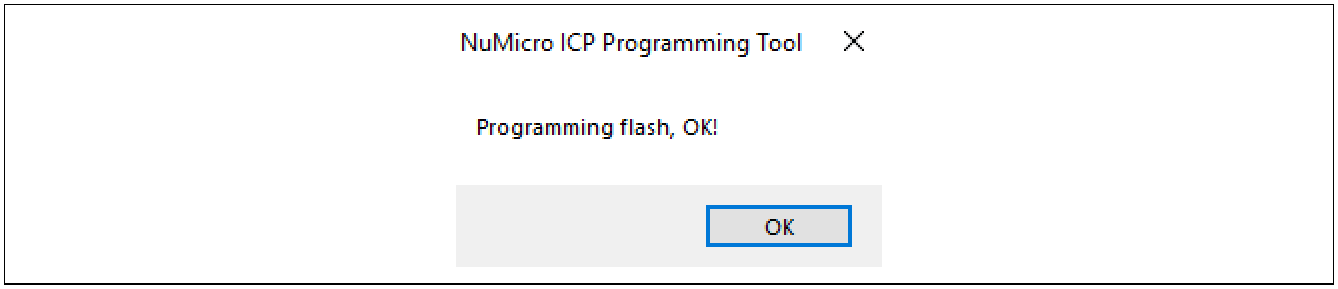
```
00000000:  02 07 B6 FF FF FF FF FF FF FF FF 02 07 91 E4 F5
00000010:  8A F5 8C 43 88 10 F5 91 43 B7 10 43 A8 02 22 FF
00000020:  FF FF FF 02 06 7C E4 F5 91 A2 AF 92 00 C2 AF 75
00000030:  C7 AA 75 C7 55 43 9F 01 A2 00 92 AF 75 C7 55 75
00000040:  C7 AA 75 AA 07 12 06 CF 12 00 0E C2 01 75 21 03
00000050:  75 22 20 C2 03 20 04 03 02 04 BD C2 AF 20 03 03
00000060:  02 01 2E 75 2A 08 AF 2A C3 EF 94 40 74 80 94 80
00000070:  40 03 02 01 0D 75 21 03 75 22 20 75 AF 21 AF 39
00000080:  EF F5 A6 E5 38 F5 A7 AF 2A 74 00 2F F5 82 E4 34
00000090:  00 F5 83 E0 F5 AE E4 F5 91 A2 AF 92 00 C2 AF 75
000000A0:  C7 AA 75 C7 55 43 A4 01 A2 00 92 AF F5 AF AF 2A
000000B0:  74 00 2F F5 82 E4 34 00 F5 83 E0 65 AE 60 02 80
```

● 8 bits
○ 16 bits
○ 32 bits

☐ APROM ☑ LDROM ☐ SPROM ☑ Config

Build: 7174r

Figure 4-9 Programming File Selection Window



NuMicro ICP Programming Tool

Programming flash, OK!

Figure 4-10 Programming Completion Window

4.3 ISP Programming

4.3.1 ISP Programming Connection

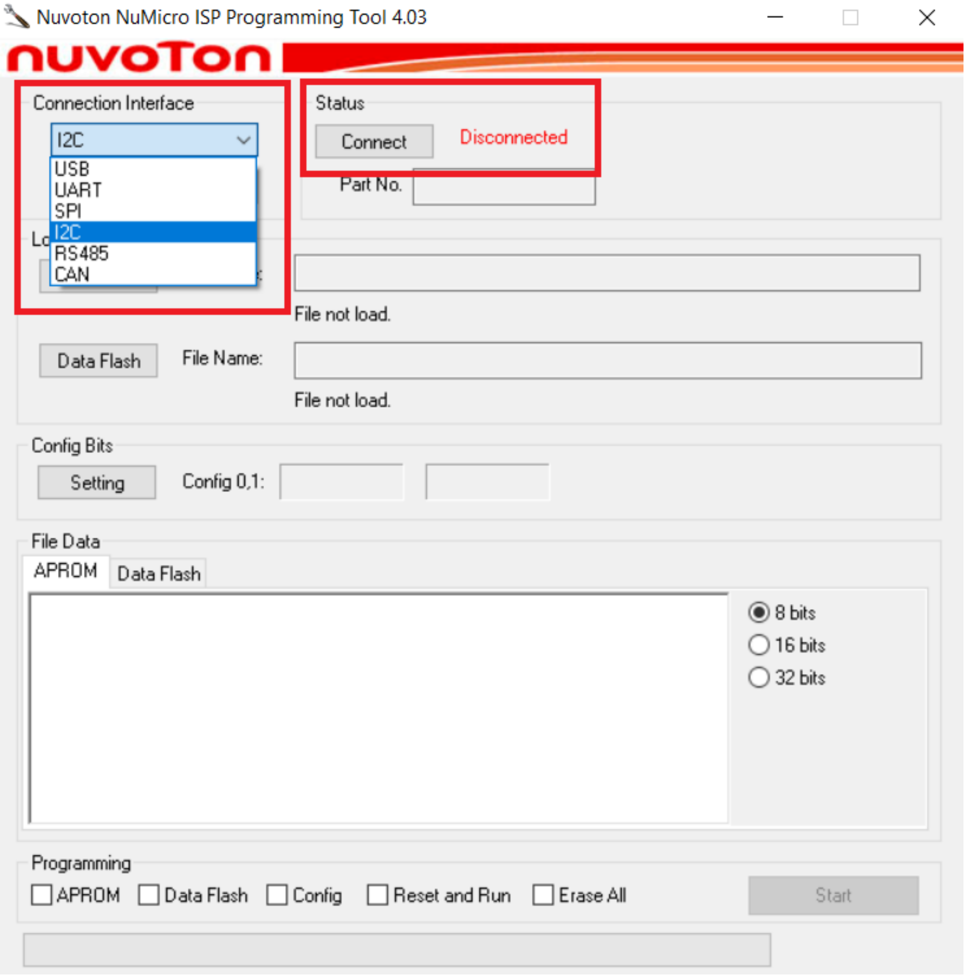
Please refer to the chapter 2.2.

4.3.2 ISP Programming Tool

(1) Download and install NuMicro® ISP Programming Tool.

(2) Open the ISP Programming Tool and select the **Connection Interface**, as shown in Figure 4-11. Please be noted that all the interfaces can be used with Nu-Link2-Pro only except for UART, which can be used with other Nu-Link debugger and programmers. The connection status is still "Disconnected" since the ISP Programming Tool has not been connected with the Nu-Link2-Pro.

Note: If the **Connection Interface** is UART, users must select the Com Port used by Nu-Link2-Pro, as shown in Figure 4-12.



Connection Interface: I2C

Status: Disconnected

File not load.

File not load.

● 8 bits

Figure 4-11 Startup Screen of ISP Tool

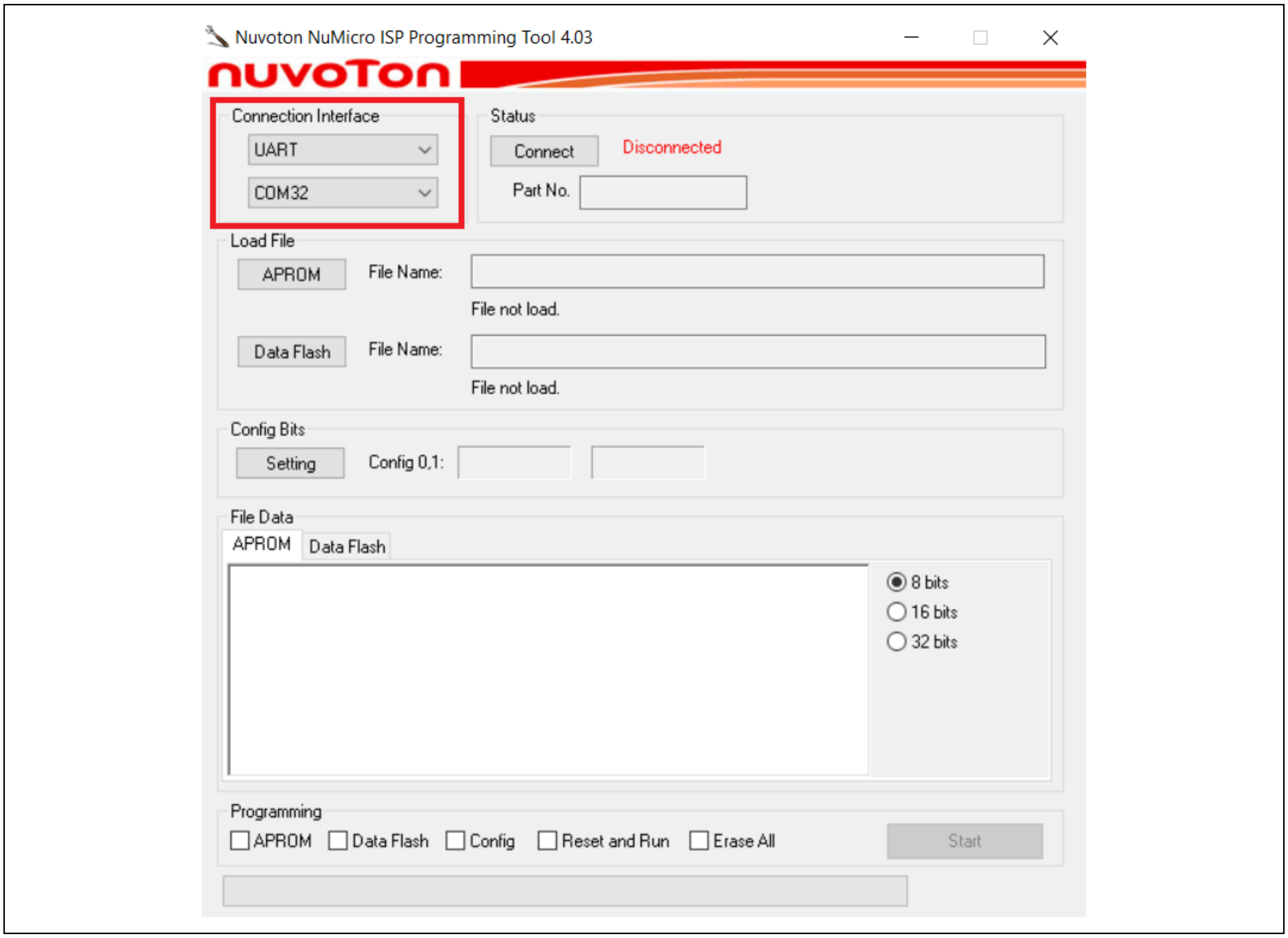
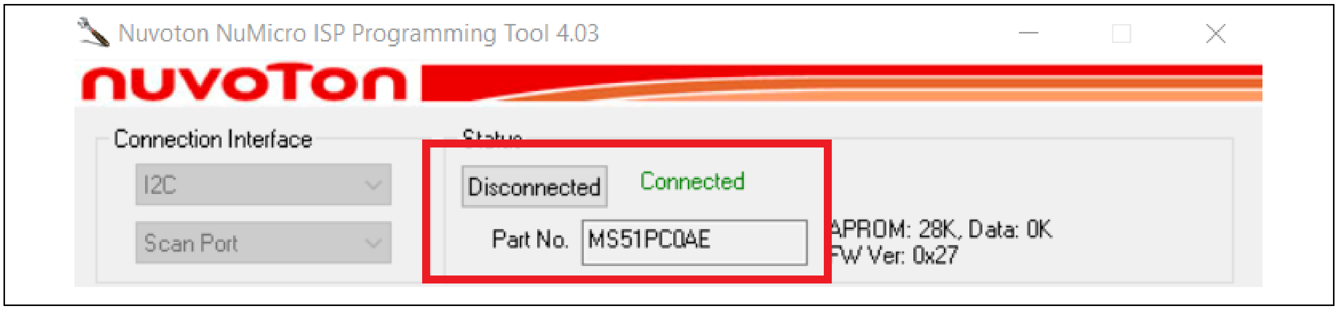


Figure 4-12 Com Port Selection of UART Interface

(3) After the **Connect** button is clicked, the ISP Tool will be connected with the Nu-Link2-Pro, and the **Connection Interface** will be detected. Figure 4-13 shows that the ISP Tool has been connected with the Nu-Link2-Pro through the I^2C interface and a target chip is detected. Meanwhile, users can start programming the target chip.



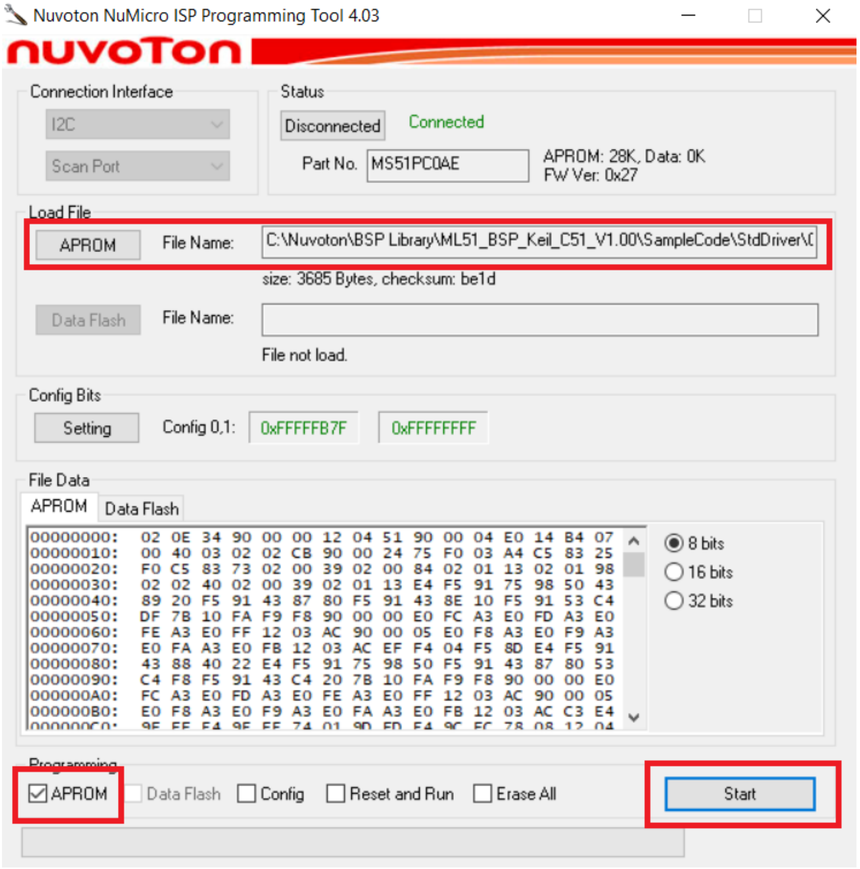
Part No. MS51PC0AE
APROM: 28K, Data: 0K
FW Ver: 0x27

Figure 4-13 Nu-Link2-Pro Connected with a Target Chip Detected

(4) As shown in Figure 4-14, select the required ".bin" file to program APROM, and tick the radio button next to the memory location aimed to program, which is APROM in this case.

(5) Click the **Start** button to start programming.

(6) As shown in Figure 4-15, click the **OK** button in the Programming Completion Window to complete the operation.



Connected

Part No. MS51PC0AE

APROM: 28K, Data: 0K
FW Ver: 0x27

File Name: C:\Nuvoton\BSP Library\ML51_BSP_Keil_C51_V1.00\SampleCode\StdDriver\

size: 3685 Bytes, checksum: be1d

File Name:

File not load.

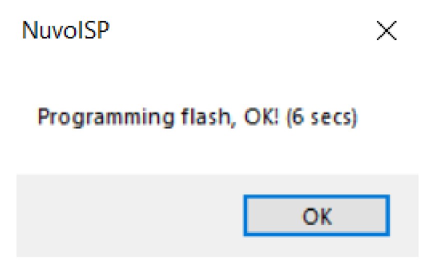
Config 0,1: 0xFFFFFB7F 0xFFFFFFFF

```
00000000:  02 0E 34 90 00 00 12 04 51 90 00 04 E0 14 B4 07
00000010:  00 40 03 02 02 CB 90 00 24 75 F0 03 A4 C5 83 25
00000020:  F0 C5 83 73 02 00 39 02 00 84 02 01 13 02 01 98
00000030:  02 02 40 02 00 39 02 01 13 E4 F5 91 75 98 50 43
00000040:  89 20 F5 91 43 87 80 F5 91 43 8E 10 F5 91 53 C4
00000050:  DF 7B 10 FA F9 F8 90 00 00 E0 FC A3 E0 FD A3 E0
00000060:  FE A3 E0 FF 12 03 AC 90 00 05 E0 F8 A3 E0 F9 A3
00000070:  E0 FA A3 E0 FB 12 03 AC EF F4 04 F5 8D E4 F5 91
00000080:  43 88 40 22 E4 F5 91 75 98 50 F5 91 43 87 80 53
00000090:  C4 F8 F5 91 43 C4 20 7B 10 FA F9 F8 90 00 00 E0
000000A0:  FC A3 E0 FD A3 E0 FE A3 E0 FF 12 03 AC 90 00 05
000000B0:  E0 F8 A3 E0 F9 A3 E0 FA A3 E0 FB 12 03 AC C3 E4
```

● 8 bits
○ 16 bits
○ 32 bits

☑ APROM ☐ Data Flash ☐ Config ☐ Reset and Run ☐ Erase All

Figure 4-14 Programming File Selection Window



Programming flash, OK! (6 secs)

Figure 4-15 Programming Completion Window

5 Summary

Nuvoton provides users all of resources to customize a total solution of ISP programming application, such as software utility with source code and the boot code of MCU as shown in Table 5-1. Nuvoton recommends setting Nu-Link2-Pro or Nu-Link-Me, depending on ISP interface, as the host. To perform ISP, the Nu-Link2-Pro has the built-in communication bridge of UART, I^{2}C, SPI, CAN and RS-485 interface, while the Nu-Link-Me has the built-in communication bridge of UART interface.

- Resource of Host side

  Download and unzip ISP programming tool file, which contains 2 folders, Application Program and User Manual, and one document. Application Program folder includes the executable software utility with source code and the boot code of MCU.

Directory	Description
Application Program	Software utility with source code and the boot code of MCU.
User Manual	This document describes the user manual of ISP programming tool

Table 5-1 ISP Programing Tool Folder Navigation

- Resource of Device side

  The project can be found under the BSP folder. Take the product ML51 as an example: the path of ISP related project is shown in Figure 5-1 ISP Project Folder Path

🗁 ML51_BSP_Keil_C51_V2.XX.XXX
  🗁 SampleCode
    🗁 ISP

Figure 5-1 ISP Project Folder Path

6 REVISION HISTORY

Date	Revision	Description
2021. 03. 30	1.00	Initial version

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.