AN10356

Entering ISP mode from user code

Rev. 03 — 13 September 2006 Application note

Document information

Info	Content
Keywords	ARM ISP, bootloader
Abstract	Entering ISP mode is normally done by sampling a pin during reset. This application note describes a method whereby ISP mode may be entered while running in user code.

Revision history

Rev	Date	Description
03	20060913	1. In section 3, an additional step was added which involved the Fractional Baud Rate generator.
02	20051025	1. Some key changes were done to the steps needed to enter ISP mode in Section 3 2. Boot flowchart was added for the LPC213x and LPC214x devices.
01	20050204	Initial version

Contact information

For additional information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: sales.addresses@www.nxp.com

1. Introduction

In-System programming (ISP) is a method of programming and erasing the on-chip flash or RAM memory using the bootloader software and a serial port. The part may reside in the end-user system. The flash bootloader provides an In-System Programming interface for programming the on-chip flash or RAM memory. This bootloader is located in the upper 8 kB of flash memory, it can be read but not written to or erased.

In many cases it may be desirable to enter ISP mode without resetting the device, while running user code. A method of doing this, which involves only a small amount of code, is described in this document.

2. LPC2000 ISP overview

The flash bootloader code is executed every time the part is powered on or reset. The loader can execute the ISP command handler or pass execution to the user application code. A LOW level, after reset, at the P0.14 pin is considered the external hardware request to start the ISP command handler. The bootloader samples this pin during reset.

Assuming that a proper signal is present on the X1 pin when the rising edge on the Reset pin is generated, it may take up to 3 ms before P0.14 is sampled and the decision on whether to continue with user code or ISP handler is made. If P0.14 is sampled LOW and the watchdog overflow flag is set, the external hardware request to start the ISP command handler is ignored. If there is no request for the ISP command handler execution (P0.14 is sampled HIGH after reset), a search is made for a valid user program. If a valid user program is found then the execution control is transferred to it. If a valid user program is not found, the auto-baud routine is invoked. As Pin P0.14 is used as the hardware request for ISP, it requires special attention. Since P0.14 is in high impedance mode after reset, it is important that the user provides external hardware (a pull-up resistor or other device) to put the pin in a defined state. Otherwise unintended entry into ISP mode may occur.

Fig 1 shows the boot sequence of the LPC2100 devices.

Fig 3 shows the boot sequence of LPC2000 devices (Bootloader revisions 1.61 and later, not applicable to the LPC210x devices).

Fig 3 shows the boot sequence of the LPC213x and LPC214x devices.

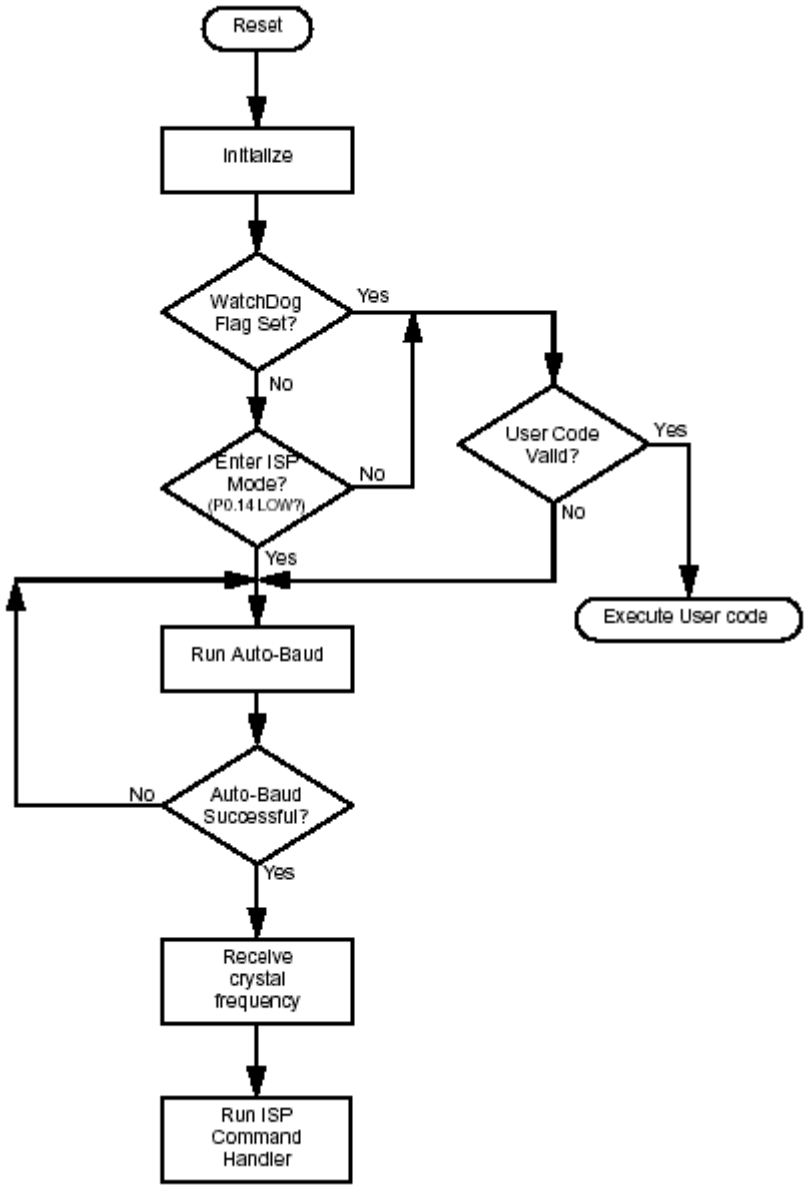

Fig 1. Boot process flowchart (LPC210x devices)

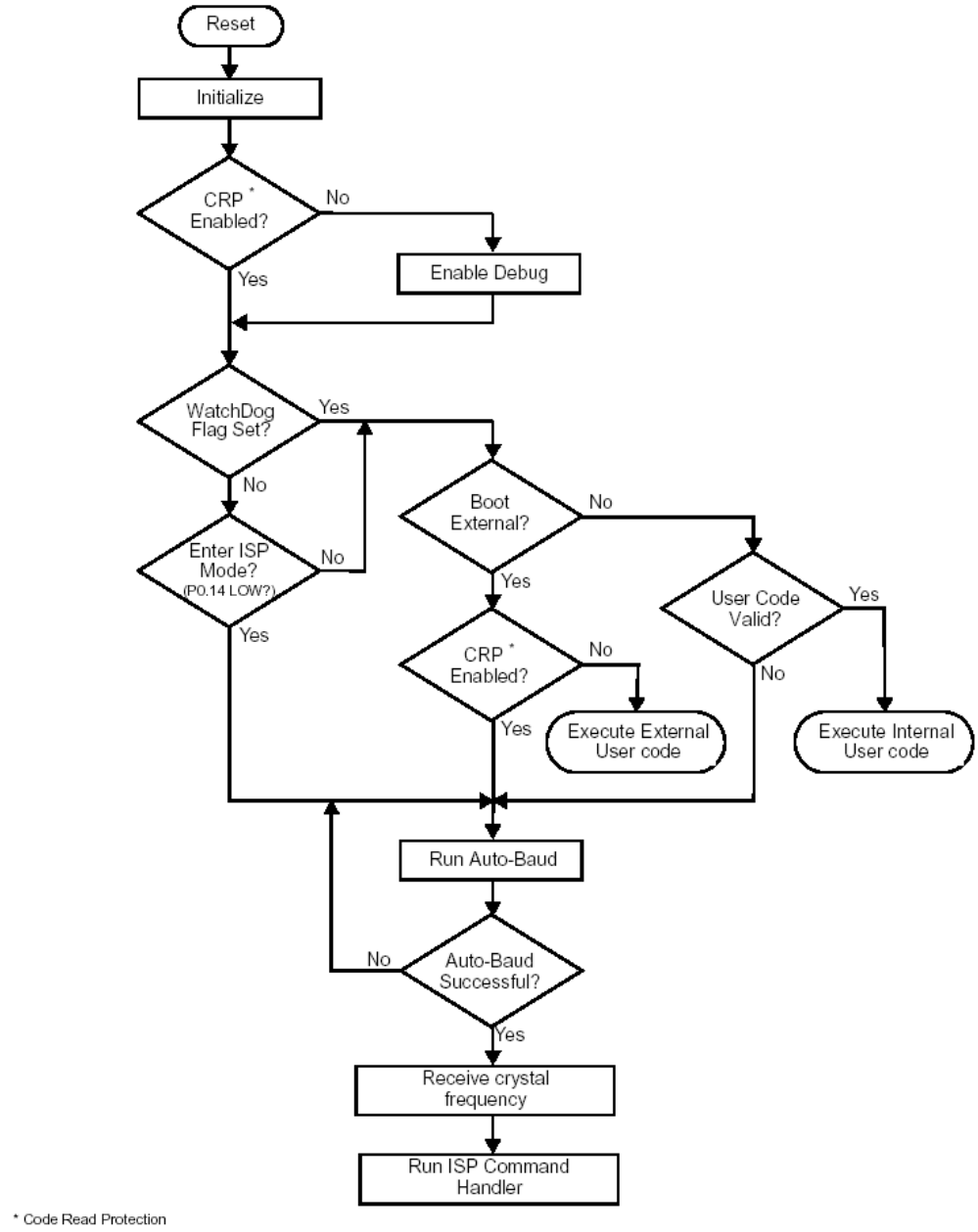

Fig 2. Boot Process flowchart (LPC2000 Bootloader revisions 1.61 and later, not applicable to the LPC210x devices)

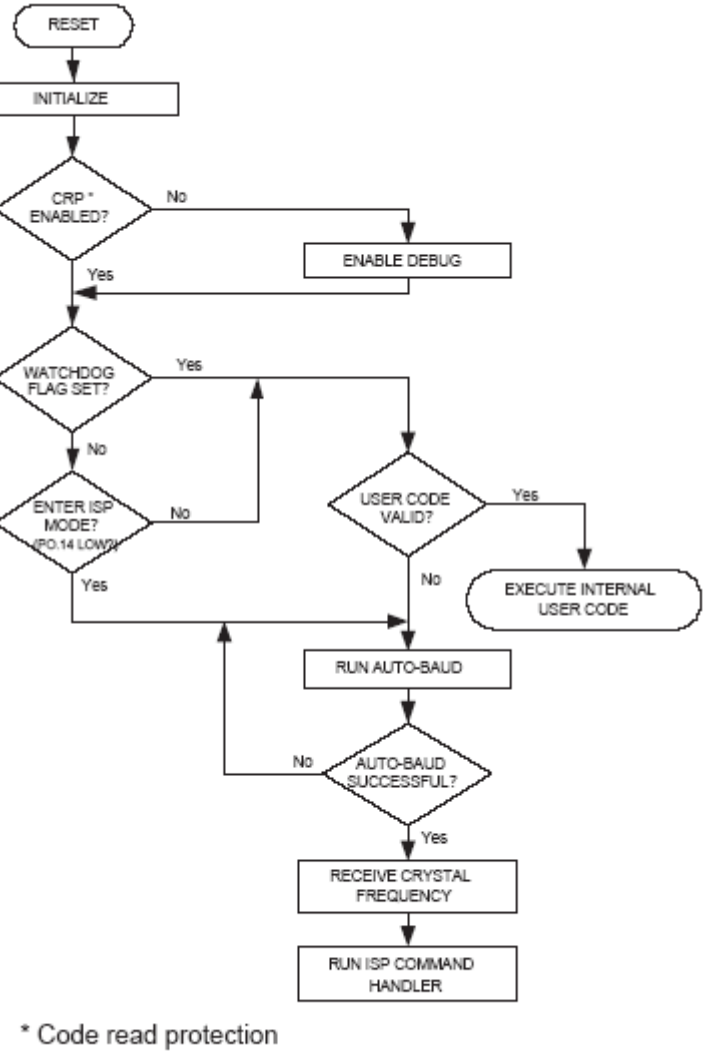

Fig 3. Boot Process flowchart (LPC213x and LPC214x devices)

3. Entering ISP Mode from User Code

The steps involved in entering ISP mode from user code are as follows:

- Configure the RXD0 pin (UART 0 receive) as input. This is done by entering the appropriate values in the PINSEL0 and IODIR0 registers.
- Configure P0.14 as an output pin.
- Clear P0.14 (Set low)
- Disable interrupts by setting the corresponding bits in the VIC Interrupt Enable Clear register (VICIntEnClear at address 0xFFFF F014) for any interrupt source that was previously enabled for interrupts.
- If the PLL is connected, disconnect it (Recommended).
- Set the Peripheral Bus Divider to ¼ if needed.
- If the UART is equipped with a Fractional Baud Rate generator then the Fractional Divider register (FDR) should be set to its reset value (This is not shown in the code below).
- Restore Timer1 to its reset state. Timer 1 is used by the ISP to auto baud.
- Re-Map the interrupt vectors to the boot block.
- Invoke the bootloader by calling a function that is located at the bootloader entry point, i.e. the reset vector at 0x00.

The code to perform the operations listed above is as follows:

```
#define MEMMAP   (*((volatile unsigned int *) 0xE01FC040))
#define IODIR0   (*((volatile unsigned int *) 0xE0028008))
#define IOCLR0   (*((volatile unsigned int *) 0xE002800C))
#define PINSEL0  (*((volatile unsigned int *) 0xE002C000))
#define VPBDIV   (*((volatile unsigned int *) 0xE01FC100))
#define PLLCON   (*((volatile unsigned int *) 0xE01FC080))
#define PLLFEED  (*((volatile unsigned int *) 0xE01FC08C))
#define VICINTENCLR  (*((volatile unsigned int *) 0Xffffff014))
#define TIMER1_PR    (*((volatile unsigned int *) 0xE000800C))
#define TIMER1_MCR   (*((volatile unsigned int *) 0xE0008014))
#define TIMER1_CCR   (*((volatile unsigned int *) 0xE0008028))

void (*bootloader_entry)(void);
unsigned long temp;

void init(void)
  {
temp = PINSEL0;
/* Connect RXD0 & TXD0 pins to GPIO */
PINSEL0 = temp & 0xFFFFFFF3;

/* Select P0.14 as an output and P0.1 as an input */
temp = IODIR0;
temp = temp | 0x4000;
temp = temp & 0xFFFFFFFD;
IODIR0 = temp;

/* Clear P0.14 */
IOCLR0 = 0x4000;
```

```
/* Disable Interrupts in the VIC*/
VICINTENCLR=0X…;


/*
   Disconnect PLL if you want to do ISP at crystal frequency.
   Otherwise you need to pass the PLL freq when bootloader goes in
   ISP mode.
   cclk = crystal when PLL is disconnected
   cclk = PLL freq when PLL is connected.

   Disconnecting the PLL is recommended.   */

   PLLCON = 0x0;
   PLLFEED = 0xAA;
   PLLFEED= 0x55;

/*
   Set the VPB divider to 1/4 if your application changes the VPBDIV value.
   The bootloader is hard-coded to use the reset value of VPBDIV register
   VPBDIV = 0x0;
*/

/* Restore reset state of Timer1 */
TIMER1_PR=0x0;
TIMER1_MCR=0x0;
TIMER1_CCR=0x0;

/* Map bootloader vectors */
   MEMMAP = 0x0;

/* Point to bootloader entry point i.e. reset vector 0x0 */

   bootloader_entry = (void (*)(void))(0x0);
}
/*
   Invoke the bootloader
   The bootloader will read pin P0.14 to detect if ISP is forced
   Since P0.14 is configured as an output and set to 0, the bootloader
   will go in ISP mode.
*/
int main(void)
  {
    init();
   while(1)
   bootloader_entry();
  }
```

4. Legal information

4.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

4.2 Disclaimers

General — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of a NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is for the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

4.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

5. Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, email to: sales.addresses@www.nxp.com

Date of release: 13 September 2006
Document identifier: AN10356_3