

Automatic Baud Rate Detection on the MSP430

Chuck Farrow

MSP430 Products

ABSTRACT

Often times, microcontrollers need to interface to a user through a serial terminal via RS-232. Automatic Baud Rate (ABR) detection allows matching of baud rates between communication terminals. This application report explains how to implement this with the MSP430F1232 USART hardware module.

1 Introduction

ABR is useful since it is possible for a microcontroller application to be connected in a system where multiple communication rates are used. ABR detection allows to detect the baud rate and self adjust to that rate. The ABR detection algorithm presented here must receive one Carriage Return <CR> character from the host when the host baud rate is between 115,200 and 14,400. And a second <CR> will be required if the host baud rate is between 9,600 and 1,200. The value of the carriage return <CR> when received will be compared to a predetermined value for a match. If a match is made the USART baud rate is adjusted accordingly. If no match is made an error code is returned. There are a variety of methods for implementing ABR, this application report will focus on this one because it requires no hardware change to the development board and briefly explain one alternative that requires a hardware change.

2 Hardware Description

The development board used to implement this application was a SoftBaugh ES1232 evaluation board. It contains an MSP430F1232 microcontroller, a 32-kHz crystal, a UART level shifter, and other circuitry. The development PC is connected to the development board via a serial cable for RS-232 communications and acts as the host for serial communications ([Figure 1](#)). Any other MSP430 family member with a hardware USART module can be used as well.

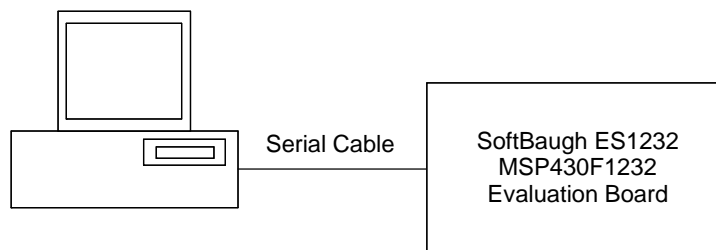


Figure 1. Demo Hardware Setup

3 Software Description

3.1 Software Overview

The demonstration software associated with this application is provided in both C and assembly language ([Table 1](#)). The main.c file is C language that can call either the C or assembly version of the autobaud algorithm. Both programs are functionally identical with the advantage of code size savings by using the assembly version.

Table 1. File Overview

FILE NAME	DESCRIPTION
Autobaud.c	Auto baud detect algorithm in C
Autobaud.s43	Auto baud detect algorithm in assembly
Main.c	Auto baud demo application in C

3.2 USART Setup

The USART is set to UART mode, for 8-bit data, one stop bit, and no parity checking. The Sub-Main System Clock (SMCLK) is used as the USART clock source and configured to operate at 1,048,576Hz (1MHz), allowing the UART to operate at baud rates up to over 230,400. For the purpose of this application report only standard rates between 1,200 and 115k will be analyzed. Although non-standard and higher rates can be utilized, they are left as an exercise for the reader. The UART is initially set to operate at 115,200 baud. The UART receive control register URCTL0 must have bit URXEIE set to allow erroneous input to trigger interrupts, as this is needed for the detection algorithm.

3.3 ABR Implementation

This section discusses how it is possible to determine the baud rate of a serial host by examining one or two characters received from that host. This application report uses a technique of examining a single carriage return to determine baud rates between 115,200 and 9,600. If the terminal is operating at a baud rate less than 9,600 then a second carriage return must be received in order to determine a baud rate between 9,600 and 1,200. The MSP430 ABR detection algorithm uses an initial baud rate setting of 115,200 to determine host baud rates between 115,200 and 9,600. And, if a second carriage return is required the ABR algorithm uses a baud rate setting of 9,600 to determine rates between 9,600 and 1,200. In both cases, the carriage return character will have a unique pattern when received. [Figure 2](#) shows how the algorithm operates.

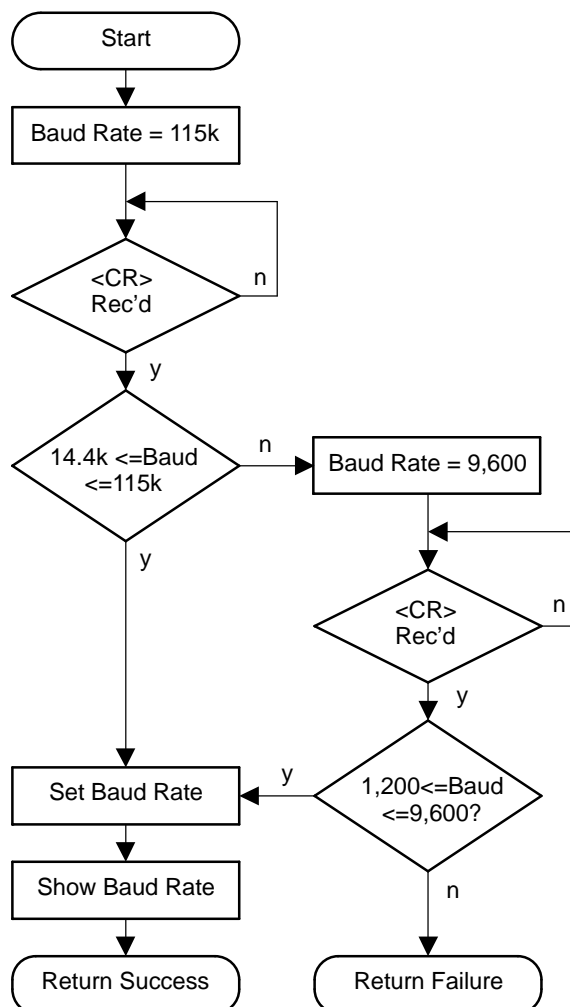


Figure 2. Software Flow Diagram

3.4 Technique for Determining Baud Rates Between 115,200 and 14,400

This section describes how the ABR detection algorithm determines the host baud rate when it is between 115,200 and 9,600. Because the MSP430 UART is setup to operate at 115,200 baud, the carriage return character will have a predictable value when received within this range of host baud rates. [Figure 3](#) shows these values. If a zero is received during this phase, it can only be determined that the host baud rate is slower than 14,400 then another carriage return <CR> will be required and this is explained in Section 3.3.2.

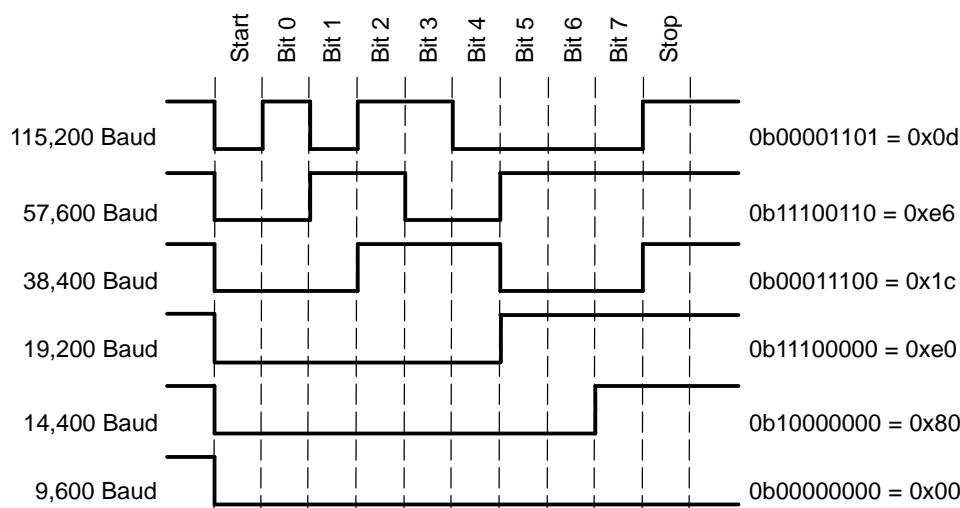


Figure 3. Bit Patterns for Baud Rates Between 115K and 9,600

3.5 Technique for Determining Baud Rates Between 9,600 and 1,200

This section describes how the ABR detection algorithm determines the host baud rate when it is between 9,600 and 1,200. To get to this portion of the algorithm the technique described in section 3.3.1 must have already determined the baud rate is slower than 14,400. If so, it will have set the baud rate generator registers to operate at a baud rate of 9,600. [Figure 4](#) shows the different patterns of the carriage return <CR> based on host baud rate setting. Once the baud rate is determined the next step is to set the MSP430 baud rate generator to the appropriate values to ensure a host baud rate match. See [Table A-1](#) in Appendix A for the appropriate register values.

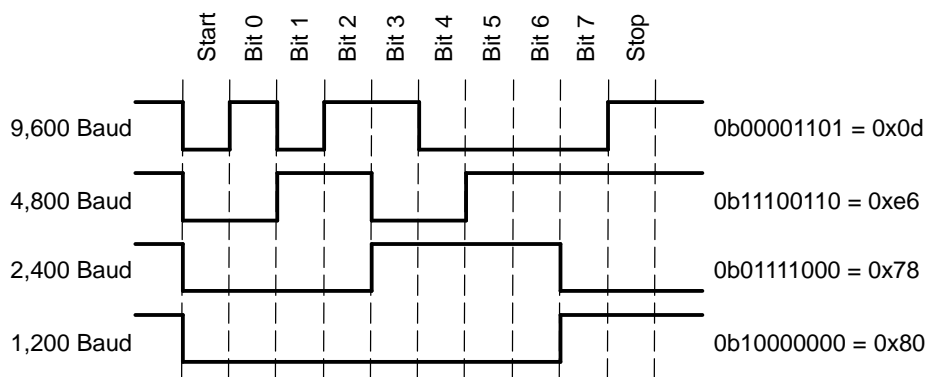


Figure 4. Bit Patterns for Baud Rates Between 9,600 and 1,200

4 Conclusion

This application report described one technique for implementing automatic baud rate detection on the MSP430 microcontroller. This technique requires the transmission of one or two carriage return <CR> characters to determine the correct baud rate. These characters will be lost due to the fact the baud rate is not correctly set during the reception of them. Although, ABR detection is applied at startup in the accompanying sample code, it could be extended and used any time the host and client determine that communication errors surpassed some threshold. This would allow for a robust application that was capable of communicating with a host that varied its baud rate between communications.

5 References

1. MSP430x1xx Family User's Guide ([SLAU049](#))
2. MSP430x12x2 Data Sheet ([SLAS361](#))
3. SoftBaugh ES1232 User's Manual (www.softbaugh.com)

Appendix A MSP430 Baud Rate Generator Registers

This section contains a table of the most common settings of the MSP4301232 USART baud rate generator registers. These values were defined based on a baud rate clock of 1,048,576Hz (1MHz). Using the formula:

$$\text{Counts} = \frac{\text{BaudRateClock}}{\text{BaudRate}}$$

Counts are converted to hex values with the MSB placed in register UBR01 and the LSB placed in register UBR00. The fractional portion is converted to a hex value based on the process defined in the MSP430x1xx Family User's Guide and placed in the modulation register UMCTL0.

Table A-1. MSP430 Baud Rate Registers for 1-MHz Clock

UBR01	UBR00	UMCTL0	COUNTS	BAUD RATE
0x00	0x09	0x10	9.10	115,200
0x00	0x12	0x84	18.20	57,600
0x00	0x1b	0x94	27.31	38,400
0x00	0x36	0xb5	54.61	19,200
0x00	0x48	0x7B	72.82	14,400
0x00	0x6d	0x44	109.23	9,600
0x00	0xda	0xaa	218.45	4,800
0x01	0xb4	0xdf	436.91	2,400
0x03	0x69	0x7b	873.81	1,200

Appendix B Alternate Method for ABR Detection

An alternate method for implementing ABR detection would be to use the on-chip timer module to determine how long it takes to receive a character from the host. For that, the receive data signal can be feed into a timer input ([Figure B-1](#)).

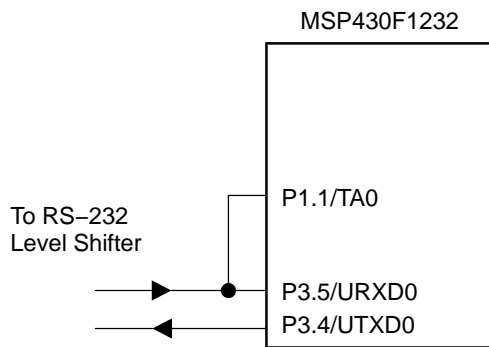


Figure B-1. Alternate ABR Using a Timer Input

With the timer module operating in capture mode, an exact time stamp of high-to-low or low-to-high transitions of the incoming receive signal can be taken. When receiving a “special” known character, such as 0x00, the time difference between the start bit and the stop bit time stamp can be calculated. With this calculation result, the exact baud rate can be determined. After that, the USART module can be configured for operation at this baud rate in order to proceed with the actual data transmission.

Any other character could be used as well for this timing analysis. For example, the high-to-low start bit transition could be used in conjunction with the low-to-high transition of the first set data bit of the particular user-defined special character.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		Applications	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

Copyright © 2004, Texas Instruments Incorporated