Implementing SDLC on the Am186™CC or Am186CH Microcontroller

Application Note

by Erwin Han

SDLC is a layer 2 communication protocol similar to the HDLC protocol. The Am186™CC and Am186CH microcontrollers support SDLC bit manipulation via their address match and address mask registers. An SDLC environment might use any of several encoding methods. Whether these microcontrollers can be used in an SDLC environment depends on the encoding method used.

OVERVIEW

This application note describes the difference between High-level Data Link Control (HDLC) and Synchronous Data Link Control (SDLC), and various methods of encoding used in serial communication. It points out design considerations for using the Am186™CC or Am186CH microcontroller in an SDLC system.

BIT MANIPULATION

The following sections introduce basic concepts of HDLC and related protocols, describe the difference between HDLC and SDLC, and explain SDLC addressing considerations.

Introduction to HDLC

In the 7-layer Open System Interconnection (OSI) model, layer 2 is the data link layer. This layer provides control between physical nodes: link initialization, flow control, and error control. One of the most common layer 2 protocols is HDLC. In HDLC, all transmissions are in frames. Actually, many other common layer 2 protocols are based on the framing structure of HDLC, like SDLC, LAPB, LAPD, etc. Figure 1 shows the frame structure of HDLC.

Flag

The flag field is defined as 7Eh (hexadecimal), which is 01111110b (binary) in serial transmission.

Zero Insertion

HDLC uses a zero insertion/deletion scheme (inserting a 0 bit after five continuous 1 bits when transmitting, then deleting the added bits when receiving) to ensure that the flag bit pattern (01111110) does not occur in the fields between flags.

Address

Because the layer 2 (data link layer) frame can be transmitted over a point-to-point link, a broadcast network, or packet switched systems, there is an address field in the frame to carry the frame's destination address. The length of this field is commonly 0, 8, or 16 bits for the "HDLC family" protocols, depending on the layer 2 protocol. For instance, HDLC and LAPD use a 16-bit address; SDLC and LAPB use an 8-bit address.

Control

The 8- or 16-bit control field provides a flow control number and defines the frame type (control or data). The exact length and use of this field depends on the protocol.

Information/Data

Data is transmitted in the information field, which can vary in length depending on the protocol. Layer 3 frames are carried in this information field.

Error Control

Error control is implemented by appending a Cyclic Redundancy Check (CRC) to the frame. All data transmitted between the opening and closing flags (excluding inserted 0s) are included in the CRC calculation. The calculated CRC is appended to the end of the frame just before the closing flag. The CRC field is 16-bits long in most protocols, but may be 32-bits long in some protocols.

Order of Transmitting

In HDLC and SDLC, the least-significant bit (LSB) of each data octet is transmitted first, and the most-significant bit (MSB) of the CRC is transmitted first.

Opening Flag	Address	Control	Information (Optional)	CRC	Closing Flag
8 bits	16 bits	8/16 bits	8N bits (N=0,1,2…)	16 bits	8 bits

Figure 1. HDLC Frame Structure

Publication# **24945** Rev: **A** Amendment/**0**
Issue Date: **April 2001**

SDLC

SDLC is a layer 2 protocol developed by IBM. It is very similar to HDLC with a slight difference.

Difference between HDLC and SDLC

To the HDLC controller of the Am186CC and Am186CH microcontrollers, SDLC is almost the same as HDLC, except for the address field. HDLC uses a 16-bit address, while SDLC uses an 8-bit address. This is not a problem in the transmitting direction, because the address field is filled by the programmer; the HDLC controller just sends it out (with zero insertion). However, in the receiving direction, the HDLC controller performs address recognition to determine whether or not to accept the current frame, so the receiver needs to know how many address bits there are.

In most cases, HDLC and SDLC are used in a point-to-point link, where destination address is not needed at all. In these cases, the address recognition function of the HDLC controller is not used for either protocol.

Address Recognition

When one of the HDLC channels in the Am186CC or Am186CH microcontroller is enabled for receiving, it waits for an opening flag. When the receiver detects the flag, it compares the frame address (the first 16 data bits following the flag) against the user-programmed addresses. Each HDLC channel has four 16-bit address match registers (the HxA0–HxA3 registers) and four corresponding 16-bit address mask registers (the HxA0MSK–HxA3MSK registers). Table 1 lists these registers for HDLC channel A. The mask register determines which bits of the first 16 data bits in the frame should be compared to the corresponding address register and which bits should be ignored.

The HDLC controller compares the received address field (16-bit) to the user-defined values (in the channel's four address registers) and masks the result with the address mask (in the corresponding address mask register). If all unmasked bits of at least one address (out of four) match, the receiver accepts the frame; otherwise, it discards the frame and starts looking for the next flag. Each HDLC channel can recognize up to four different mask and address sets. Each set can select either one address or multiple addresses, using either 8-bit or 16-bit addressing.

Table 1. HDLC Channel A Address Registers

Offset	Mnemonic	Register Name
22h	HAA0	HDLC Channel A Address 0
24h	HAA0MSK	HDLC Channel A Address Mask 0
26h	HAA1	HDLC Channel A Address 1
28h	HAA1MSK	HDLC Channel A Address Mask 1
2Ah	HAA2	HDLC Channel A Address 2
2Ch	HAA2MSK	HDLC Channel A Address Mask 2
2Eh	HAA3	HDLC Channel A Address 3
30h	HAA3MSK	HDLC Channel A Address Mask 3

Note: *Register bits 7–0 correspond to the first byte received, and bits 15–8 correspond to the second byte received.*

The HDLC controller can also recognize broadcast address frames (all bits = 1), if one address match register is written with all bits set to 1.

For 8-bit addresses of SDLC, just mask (clear) the eight high-order bits in one of the HxA0MSK–HxA3MSK registers.

16-Bit Address Recognition Example

To accept a frame that begins with the three bytes 7Eh (flag), 5Bh, and ACh, using 16-bit address recognition, one of the HxA0–HxA3 registers should contain AC5Bh, and the corresponding HxAnMSK register should contain FFFFh.

The configuration in Figure 2 recognizes:

- One 16-bit address: AC5B
- The 16-bit broadcast address: FFFF
- A set of 16-bit addresses: xx5C (x=0–F)
- A set of 16-bit addresses: AC60–AC6F

Frame with 16-bit Address

Flag 7E	Address 5B	Address AC	Control 9C	Etc.

Match and Mask Register Contents

HxA0	AC5B	HxA2	AC5C
HxA0MSK	FFFF	HxA2MSK	00FF
HxA1	FFFF	HxA3	AC6x
HxA1MSK	FFFF	HxA3MSK	FFF0

Figure 2. 16-Bit Address Matching Example

8-Bit Address Recognition Example

To accept a frame that begins with 7Eh (flag) and 5Bh, using 8-bit address recognition, the low byte of one HxA0–HxA3 register should contain 5Bh, and the corresponding HxAnMSK should contain 00FFh.

The configuration shown in Figure 3 recognizes:

- One 8-bit address: 5B
- A set of 8-bit address: 50–5F

Frame with 8-bit Address

Flag 7E	Address 5B	Control AC	Etc.

Match and Mask Register Contents

HxA0	xx5B	HxA2	xxxx
HxA0MSK	00FF	HxA2MSK	0000
HxA1	xx5x	HxA3	xxxx
HxA1MSK	00F0	HxA3MSK	0000

Figure 3. 8-Bit Address Matching Example

Note: *If one of the HxA0–HxA3 registers contains xx5Bh, and the corresponding HxAnMSK contains 00FFh, the HDLC controller accepts any frame starting with 7E (Flag), 5Bh. This might be either a 16-bit address xx5Bh or an 8-bit address 5Bh. Because the content of the frame (starting from the byte after Flag) is saved in the receiving buffer, the programmer will know whether to handle it as 16-bit address or 8-bit address.*

Control

In some SDLC controllers, there is logic to automatically handle certain types of frames, as defined by the bits in the control field. The HDLC controllers in the Am186CC and Am186CH microcontrollers do not support automatic handling; it must be done by software.

ENCODING

In serial communication, there are many ways to encode the data, such as NRZ, NRZI, FM0, FM1, Manchester, etc. Examples of some different encoding methods are shown in Figure 4.

NRZ (Non-Return to Zero)

In NRZ encoding, a 1 bit is represented by a High level and a 0 bit is represented by a Low level. In this encoding method, only a minimal amount of clocking information is available in the data stream in the form of transitions on bit-cell boundaries. In an arbitrary data pattern, this may not be sufficient to generate a clock for the data from the data itself.

NRZI (Non-Return to Zero Inverted)

In NRZI encoding, a 1 bit is represented by no change in the level, and a 0 bit is represented by a change in the level. As in NRZ, only a minimal amount of clocking information is available in the data stream, in the form of transitions on bit cell boundaries. In an arbitrary data pattern, this may not be sufficient to generate a clock for the data from the data itself. In the case of HDLC/SDLC mode, where the number of consecutive 1 bits in the data stream is limited (by zero insertion), a minimum number of transitions to generate a clock is provided.

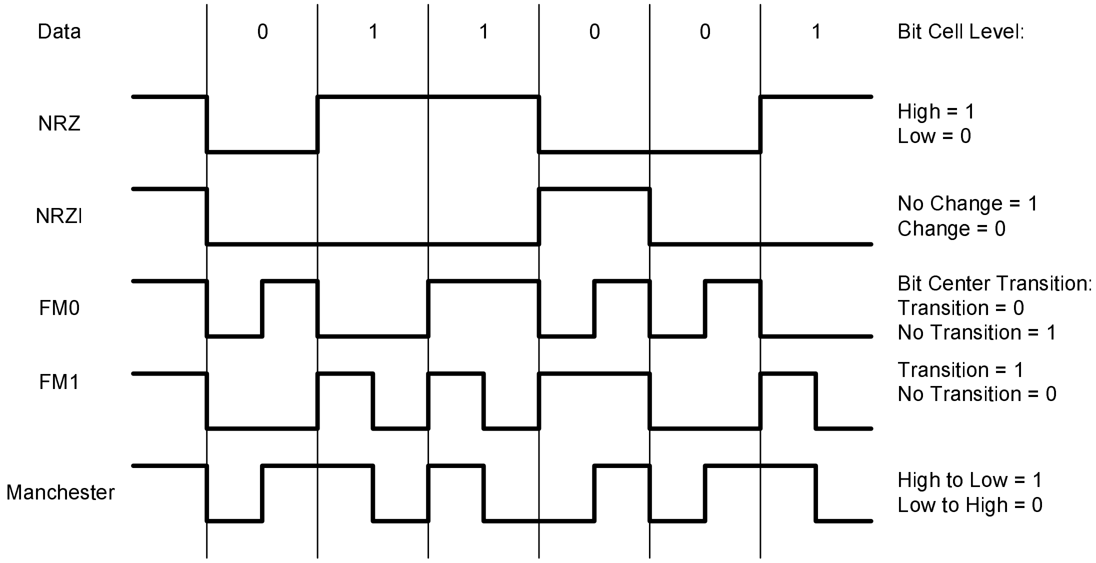

Figure 4. Data Encoding

FM0 (Biphase Space)

In FM0 encoding, also known as biphase space, a transition is present on every bit cell boundary, and an additional transition may be present in the middle of the bit cell. In FM0, a 1 bit is sent as no transition in the center of the bit cell, and a 0 bit is sent as a transition in the center of the bit cell. FM0 encoding data contains sufficient information to recover a clock from the data.

FM1 (Biphase Mark)

In FM1 encoding, also known as biphase mark, a transition is present on every bit cell boundary, and an additional transition may be present in the middle of the bit cell. In FM1, a 0 bit is sent as no transition in the center of the bit cell, and a 1 bit is sent as a transition in the center of the bit cell. FM1 encoded data contains sufficient information to recover a clock from the data.

Manchester

Manchester encoding always produces a transition at the center of the bit cell. If the transition is High-to-Low, the bit is a 1; if the transition is Low-to-High, the bit is a 0. Manchester encoded data contains sufficient information to recover a clock from the data. Manchester encoding is used in 10-Mbit/s Ethernet.

Digital Phase-Locked Loop (DPLL)

Many communication controllers (like the Am85C30, MC68360, and all 10 Mbit/s Ethernet controllers) have a DPLL that can be used to recover clock information from a data stream with NRZI, FM or Manchester encoding. The data stream encoded in these ways contains sufficient information to recover a clock from the data.

Advantage and Drawback

The advantage of encoding mechanisms that include sufficient clock information (e.g., NRZI, FM, and Manchester) is that clock information can be recovered from the data stream so a separate clock is not needed. Therefore, the number of wires for transmission is minimized. However, FM and Manchester encoding have a drawback in that the worst-case signaling rate is twice the data rate.

Encodings Supported by the Am186™CC and Am186CH Microcontrollers

The Am186CC and Am186CH microcontrollers support NRZ and NRZI encoding. They do not support FM0, FM1, and Manchester encoding. These microcontrollers do not have a DPLL, therefore they cannot recover clock information from an NRZI data stream. When NRZI is used and there is no separate clock accompanying the data stream, these microcontrollers need an external DPLL to retrieve a clock signal from the received data. This clock will be used for both receiving and transmitting.

SUMMARY

The HDLC controller in the Am186CC and Am186CH microcontrollers supports all the bit manipulation of the SDLC protocol. Whether or not these microcontrollers can be used in an SDLC environment depends on what method of encoding is used.

Table 2. Encoding Support

Encoding Method	Supported by Am186™CC and Am186CH Microcontrollers
NRZ	Yes
NRZI with separate clock	Yes
NRZI without separate clock	Yes, but only with external DPLL
FM0 or FM1	No
Manchester	No

Trademarks

AMD, the AMD logo, and combinations thereof, and Am186 are trademarks of Advanced Micro Devices, Inc.

Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Disclaimer

The contents of this document are provided in connection with Advanced Micro Devices, Inc. (“AMD”) products. AMD makes no representations or warranties with respect to the accuracy or completeness of the contents of this publication and reserves the right to make changes to specifications and product descriptions at any time without notice. No license, whether express, implied, arising by estoppel or otherwise, to any intellectual property rights is granted by this publication. Except as set forth in AMD's Standard Terms and Conditions of Sale, AMD assumes no liability whatsoever, and disclaims any express or implied warranty, relating to its products including, but not limited to, the implied warranty of merchantability, fitness for a particular purpose, or infringement of any intellectual property right.

AMD's products are not designed, intended, authorized or warranted for use as components in systems intended for surgical implant into the body, or in other applications intended to support or sustain life, or in any other application in which the failure of AMD's product could create a situation where personal injury, death, or severe property or environmental damage may occur. AMD reserves the right to discontinue or make changes to its products at any time without notice.