AP-722

APPLICATION NOTE

Interfacing an Intel 82527 Serial Communications Controller to a 68HC11

GREG SCOTT
TECHNICAL MARKETING ENGINEER

October 1995

Order Number: 272762-001

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

INTERFACING AN INTEL 82527 SERIAL COMMUNICATIONS CONTROLLER TO A 68HC11

CONTENTS

1.0 INTRODUCTION

The purpose of this application note is to describe one method of interfacing a Motorola 68HC11Ax microcontroller with an Intel 82527 CAN controller. This description includes a demonstration that all AC timing specifications are satisfied and a brief description of the 82527 interface modes and clocking structure.

1.1 Interface Suggestions

- The 82527 should be used in mode 2.
- The 82527 matches 68HC11 microcontroller pin for pin on: AS, R/W#, E, and AD7–AD0.
- The 82527 INT# pin is tied to the 68HC11 microcontroller IRQ# pin.
- The 82527 RESET# pin is tied to a 68HC11 port pin or reset circuit.
- The CS# signal may be generated by decoding the upper address bits from the 68HC11 microcontroller.

1.2 Top Four Issues

- For bus frequencies above 1.43 MHz, the double read operation is required for reading low speed registers.
- The t_{AVSL} and t_{CLSC} time specifications for mode 2 of the 82527 are 7.5 ns and 10 ns, respectively. All 82527 controllers meet these timing specifications. This change will be reflected in the next data sheet (release date Q1'96).
- 82527 RESET# pin must be asserted low for 1ms minimum.
- Default condition of 82527 for mode 2 sets MCLK = SCLK/2, this must be set to MCLK = SCLK following a reset.

2.0 82527 CPU INTERFACE MODES

The 82527 supports six CPU interface modes allowing users to connect the 82527 to host-CPU's of various architectures. The CPU interface modes are:

- 8-bit Intel multiplexed (mode 0)
- 16-bit Intel multiplexed (mode 1)
- 8-bit non-Intel multiplexed (mode 2—AS, E, R/W#)
- 8-bit non-multiplexed synchronous (mode 3)
- 8-bit non-multiplexed asynchronous (mode 3)
- Serial (SPI compatible)

When interfacing the 68HC11 microcontroller with multiplexed bus to the 82527, 8-bit multiplexed mode (Mode 2) of the 82527 is used.

3.0 82527 CLOCKING STRUCTURE

The operation of the 82527 is controlled by two clocks: the system clock (SCLK) and the memory clock (MCLK). The SCLK is derived from the external oscillator, while the MCLK is based off the frequency of the SCLK. The bit timings for all CAN bus communications are based on the frequency of the SCLK, while the MCLK provides clocking for all read and write operations to the 82527 RAM via the 68HC11/82527 interface.

The frequency of the SCLK may be equal-to or one-half the external oscillator frequency and is defined by the value of the DSC bit in the CPU interface register. The maximum frequency of the SCLK is 10 MHz as specified in the 82527 datasheet (Order Number 272250). An 8 MHz SCLK frequency is typically sufficient to interface the 82527 to a 1 Mbit/sec CAN bus.

The frequency of the MCLK may be equal-to or one-half the frequency of the SCLK, and is defined by the value of the DMC bit in the CPU interface register. The maximum frequency of the MCLK is 8 MHz, as specified in the 82527 datasheet (Order Number 272250). The default condition of the CPU interface following a reset is 61h. This default condition configures the SCLK to XTAL/2 and the MCLK to SCLK/2.

Double Read Operation

The 82527 contains two types of registers: high-speed registers (locations 02H, 04H, and 05H) and normal or low-speed registers (all registers except 02H, 04H, and 05H). Read and write operations to the low-speed registers occur over a synchronous internal bus which is clocked by the MCLK. High-speed registers 02H, 04H and 05H are decoupled from the internal bus, allowing them to be accessed more quickly by the 68HC11. High-speed read registers 04H and 05H are implemented for the double-read operation. The double-read operation is used for interfacing the 82527 with faster CPUs that do not allow for long access time. Interfacing the 68HC11 microcontroller operating at greater than 1.43 MHz to the 82527 requires the use of the double read operation. A brief explanation of a double read operation is contained in Section 5.1. For a more detailed explanation of the double read operation, please refer to the 82527 Architectural Overview (Order Number 272410).

4.0 INTERFACING SCHEMATICS FOR MODE 2

Figure 4-1 demonstrates a minimal hardware interface between the 68HC11 with multiplexed external bus and the 82527.

The CS# signal is derived by inverting the address line, A15. Both devices are clocked by Quartz crystals; consult the crystal manufacturer specifications for proper load capacitance. For the lowest access time of the 82527 use either a 16 MHz or an 8 MHz crystal. If the 16 MHz crystal is used, then the SCLK = XTAL/2. If the 8 MHz crystal is used, then the SCLK = XTAL. The SCLK is programmed by writing to the CPU Interface Register (location 02H). For more information on this register refer to the architectural overview (Order Number 272410). The RESET# signal for the 82527 may be generated using a port pin on the 68HC11 microcontroller, or may be derived by an RC network. The RESET# pin on the 82527 must be asserted to V_{IL} or less for a minimum of 1 ms after V_{CC} in the operation range to guarantee a proper device reset.

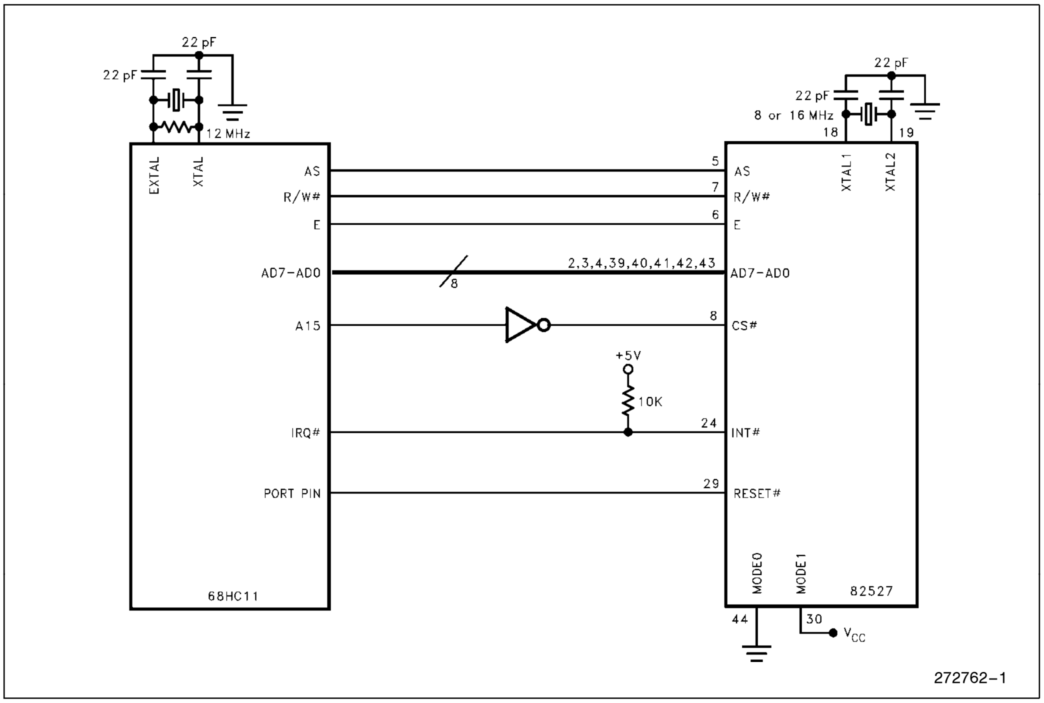

Figure 4-1. Interface Scheme

The CS# signal may be generated by a PAL decoding the upper address lines. The following equation is used for calculating the maximum propagation delay that the PAL must meet. The CS# signal must be generated 131 ns after a valid address is driven on the bus. The 82527 requires the CS# signal to be valid 20 ns before AS goes low (t_{CLSL} = 20 ns, refer to Note 5 of the timing table). The 68HC11 microcontroller sends a valid address 151 ns prior to AS falling (t_{AVSL} = 151 ns for a bus frequency of 1 MHz). The equation to calculate the time to generate the 82527 chip select is:

$$CS\# \text{ generation} < t_{AVSL} - t_{CLSC}$$
$$CS\# \text{ Generation} < 131 \text{ ns}$$

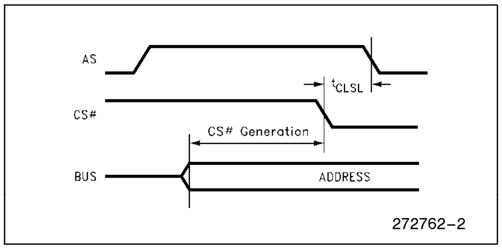

Fig 4-2. 82527 CS# Setup Timing Requirements

5.0 TIMING CONSIDERATIONS

Table 5-1. Critical Timing

Symbol	Parameter	82527 @ 8 MHz[1]	68HC11Ax @ 1 MHz	68HC11Ax @ 3 MHz
t_{AVSL}	Address Valid to AS Low	33[(4)]	151	13
t_{SLAX}	Address Hold after AS Low	20	95.5	26
t_{ELDZ}	Data Float after E Low	45 max	145.5 max	51 max
t_{EHDV}	E High to Data Valid for HSR[(2)]	45 max	442 max	111 max
t_{EHDV}	E High to Data Valid for LSR[(3)]	287.5 max	442 max	111 max
t_{QVEL}	Data Setup to E Low	30	30	30
t_{ELQX}	Input Data Hold after E Low	20	95.5	26
t_{EHEL}	E High Time	45	472	141
t_{SHSL}	AS High Time	30	221	63
t_{RSEH}	Setup time of R/W# to E High	30	281.5	54
t_{SLEH}	AS Low to E High	20	115.5	31
t_{CLSL}	CS# Low to AS Low	20[(5)]		
t_{ELCH}	E Low to CS# High	0		

NOTES:
1. All timings are given in ns and are the minimum specification unless otherwise noted.
2. Time valid for reading the High Speed Read registers (HSR). This time specification also applies to the double read operation.
3. Time valid for a read cycle without a previous write of the Low Speed Registers (LSR).
4. t_{AVSL} is 7.5 ns, this will be reflected in the next revision of the 82527 datasheet in Q1'96.
5. t_{CLSC} is 10 ns, this will be reflected in the next revision of the 82527 datasheet in Q1'96.

5.1 Read Timings

For bus frequency above 1.43 MHz (this corresponds to a crystal frequency of 5.72 MHz), double read operations are required. A double read operation is implemented by first addressing the desired low speed register then addressing the high speed read register (location 04H). High-speed registers can be accessed within a single read operation. Note that a double read operation is only required for reading a low speed register. For more information on the double read operation refer to the 82527 architectural overview (Order Number 272410). The condition of a "read cycle with a previous write cycle" as specified in the 82527 datasheet never occurs using the 68HC11 microcontroller, therefore should be ignored.

For bus frequencies below 1.43 MHz, double reads are not required. The reading of a low-speed register may be accomplished by using a single read, for either a low-speed or high-speed register. The condition of a "read cycle with a previous write cycle" as specified in the 82527 datasheet never occurs using the 68HC11 microcontroller, therefore should be ignored.

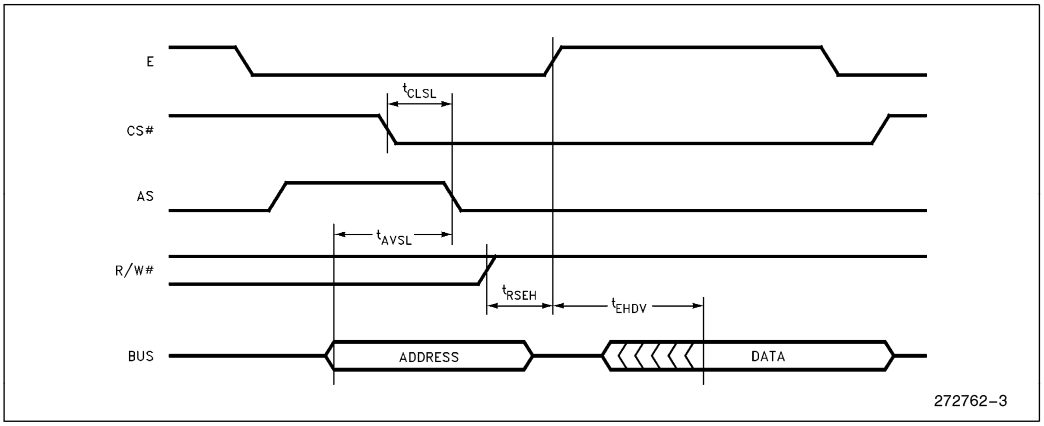

Figure 5-1. Bus Timing Diagram for a Read Cycle

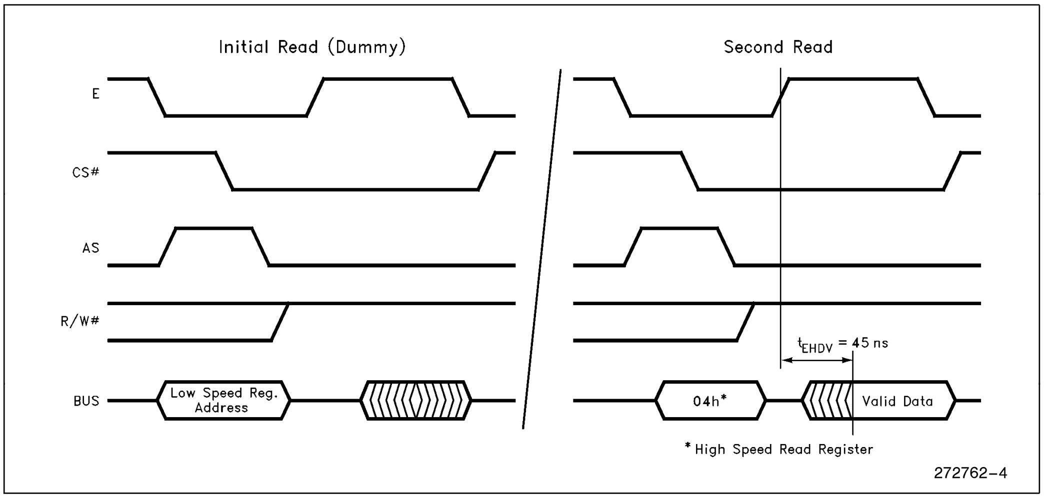

Figure 5-2. Bus Timing Diagram for a Double-Read Operation

5.2 Write Timings

The timing comparisons for a write operation are valid. No special operations are needed. The condition of "write cycle with a previous write cycle" as specified in the 82527 datasheet is not valid.

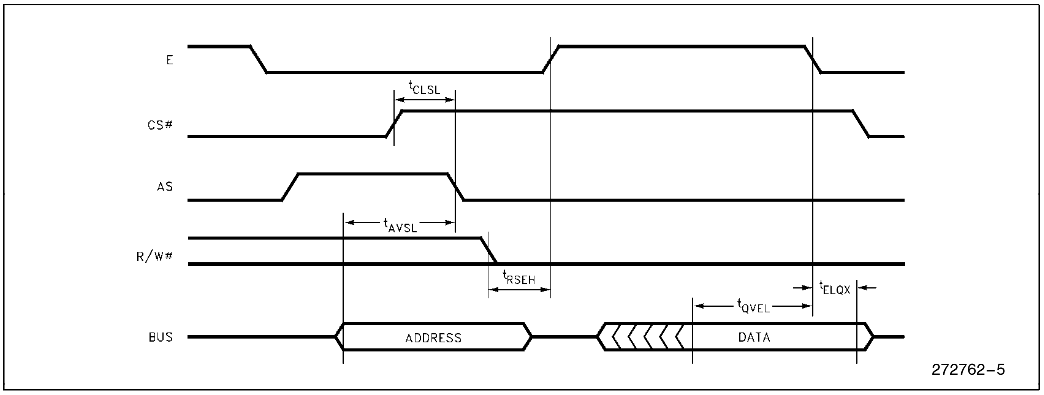

Figure 5-3. Bus Timing Diagram for a Write Cycle

6.0 C-PROGRAM FOR INITIALIZING THE 82527

This program initializes the 82527 CAN controller. For specific details on the functionality of the 82527 registers please refer to the Architectural Overview (Order Number 272410).

- Disables CLKOUT.
- Sets SCLK = XTAL/2 and MCLK = SCLK.
- Sets CAN bus rate to 250 kBits/s.
- Sets message 1 to transmit.
- Sets message 2 to receive.
- Assumes a transceiver is used.

NOTE:
This code was compiled and tested using the BSO compiler for the MCS®96 controller.

```
#define CAN 0x8000                          /* Defines the starting address of
                                               the 82527 chip. */
main()
{
   int t, x;
   unsigned char *cr, *cir, *bcr, *bt0;
   unsigned char *btl, *contr0, *contrl;
   unsigned char *gm, *mcr, *arb;           /* Initializes the counter and
                                               pointer variables. */
   cir = (unsigned char*)CAN + 2;
   *cir = 01;                               /* Set the CPU Interface Register
                                               to 40: SCLK = XTAL/2,
                                               MCLK = SCLK, and disables
                                               the CLKOUT signal. */
   cr = (unsigned char*)CAN;
   *cr = 0x41 || *cr;                       /* Sets the CCE (Change
                                               Configuration Enable) bit in the
                                               Control Register. */
   bcr = (unsigned char*)CAN + 0x2f;
   *bcr = 0x48;                             /* Sets the Bit Configuration
                                               Register to 48. This bypasses
                                               the input comparator, sets
                                               logical one as recessive, and
                                               disables the TX1 driver. DcR0
                                               and DcR1 are don't cares (set
                                               to 0 in this case) */
   bt0 = (unsigned char*)CAN + 0x3f;
   btl = (unsigned char*)CAN + 0x4f;
   *bt0 = 0x41;
   *btl = 0x67;                             /* Defines the CAN bus frequency as
                                               250 kBits/s and the sampling
                                               mode. */
   *cr = 01;                                /* Clears the CCE bit, preventing
                                               write access of configuration
                                               registers */
   for (t = 0x10; t <= 0xF0; t = t + 0x10)
   {
      contr0 = t + (unsigned char*)CAN;
      contrl = contr0 +1;
      *contr0 = 0x55;
      *contrl = 0x55;
```

```
    }                                       /*This loop resets Control
                                              Register 0 and 1. */
    for (t = 0x06; t <= 0x0b; t++)
    {
       gm = (unsigned char*)CAN + t;
       *gm = 0xff;
    }                                       /*This loop sets the Global Mask
                                              (Standard and Extended) to must
                                              match. */
    mcr = (unsigned char*)CAN + 0x16;
    *mcr = 0x8c;
    mcr = (unsigned char*)CAN + 0x26;
    *mcr = 0x84;                            /*Sets the Message Configure
                                              Registers for message 1 and 2.
                                              This sets message 1 to transmit
                                              eight bytes using an extend
                                              identifier and sets message 2
                                              to receive eight bytes using an
                                              extended identifier. */
    for (t = 0x10; t <= 0x20; t = t + 0x10)
    {
       for (x = 2; x<= 5; x++)
       {
          arb = (unsigned char*)CAN + t + x;
          *arb = 0xc8;
       }
    }                                       /*Loads $C8 into the arbitration
                                              registers */
    *cr = 00;                               /*Takes the 82527 out of the
                                              initialization mode. */
    for (t = 1;t <=4; t++)
       t = 2;
}
```