STE10/100A - EEPROM ACCESS AND PROGRAMMING

1.0 EEPROM

There are four EEPROM interface pins
(Can refer to "CSR9 SPR - Serial Port Register" on page 22/66 of the STE10/100A Data Sheet)

- EEPROM data out, EDO (pin 72)
- Serial ROM data in, EDI (pin 73)
- Serial ROM clock, ECK (pin 74)
- Serial ROM chip select, EECS (pin 76)

The STE10/100A is designed to support serial interface to a 93C46 EEPROM.
The EEPROM has three types of information:

- Information that is used by the STE10/100A
- Information that can be used by the STE10/100A driver
- CIS data

The information that is used by the STE10/100A is located in the ID block. These blocks are automatically read by the STE10/100A without software involvement. The ID block is read upon a hardware reset or when the STE10/100A transitions from the D3 power state to the D0 power state.
The ID block is located at the top of the EEROM, beginning in address 0.

The STE10/100A driver accesses the EEPROM through CSR9. The access sequences and timing are handled by the software. The EEPROM operations in this method can be read, write, or erase. The read and write operations in this method are described in the following sections. The erase operation is handled very similarly to the read and write operations.

2.0 Read Operation

Read operations consist of three phases:
1. Command phase—3 bits (binary code of 110)
2. Address phase—6 bits for 256-bit to 1 Kb ROMs, 8 bits for 2 Kb to 4 Kb ROMs.
3. Data phase—16 bits

Figure 1 and Figure 2 show a typical read cycle that describes the action steps that need to be taken by the driver to execute a read cycle. The timing (listed on the right side of the figures) specifies the minimum time that the driver must wait before advancing to the next action. During both the address phase in Figure 1 and data phase in Figure 2, 1 bit is handled during each phase cycle. Therefore, the address phase should be repeated 6 or 8 times depending on the address length and data phase should be repeated 16 times. Note the value DX is the current data bit.

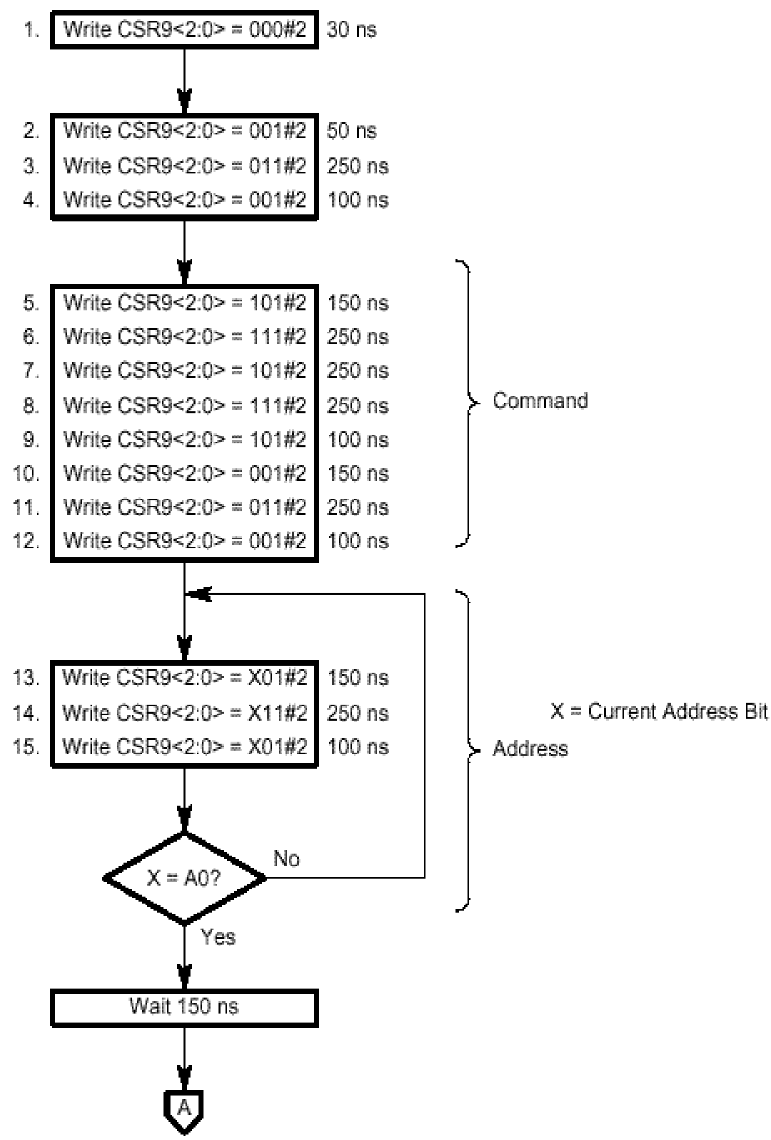

Figure 1. Read Cycle (1 of 2)

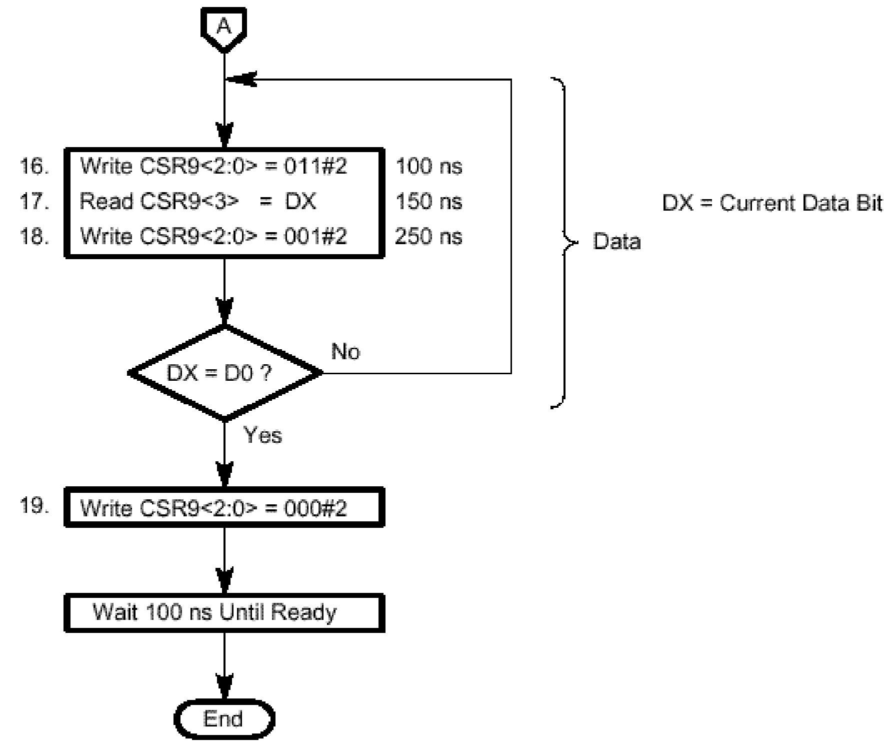

Figure 2. Read Cycle (2 of 2)

The read operation timing of the address and data are showed below:

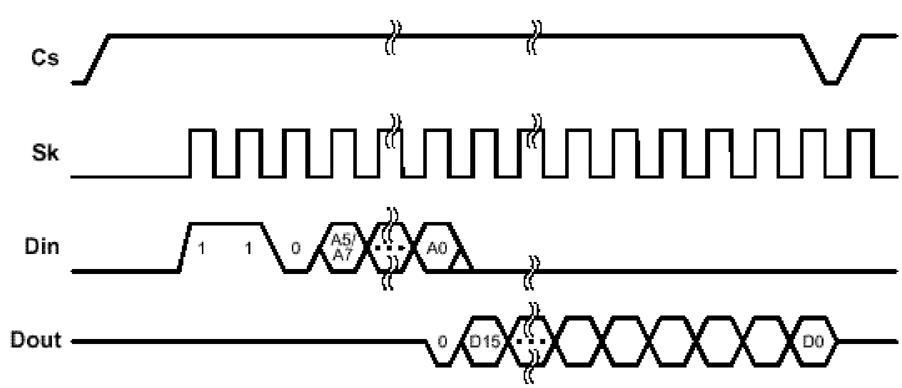

Figure 3. Read Operation

3.0 Write Operation

Write operations consist of three phases:

1. Command phase—3 bits (binary code of 101)
2. Address phase—6 bits for 256-bit to 1Kb ROMs, 8 bits for 2Kb to 4Kb ROMs.
3. Data phase—16 bits

Figure 4 and Figure 5 show a typical write cycle that describes the action steps that need to be taken by the driver to execute a write cycle. The timing (listed on the right side of the figures) specifies the minimum time that the driver must wait before advancing to the next action.

During both the address phase in Figure 4 and the data phase in Figure 5, 1 bit is handled during each phase cycle. Therefore, the address phase should be repeated 6 or 8 times depending on the address length and the data phase should be repeated 16 times.

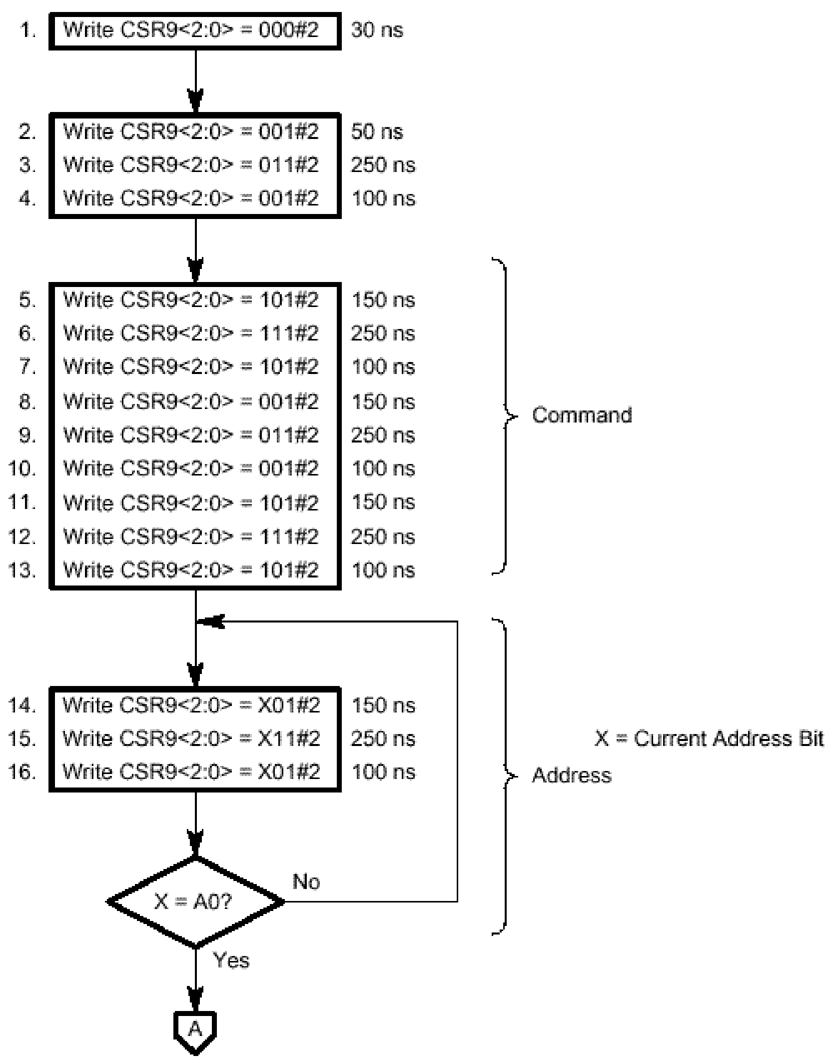

Figure 4. Write Cycle (1 of 1)

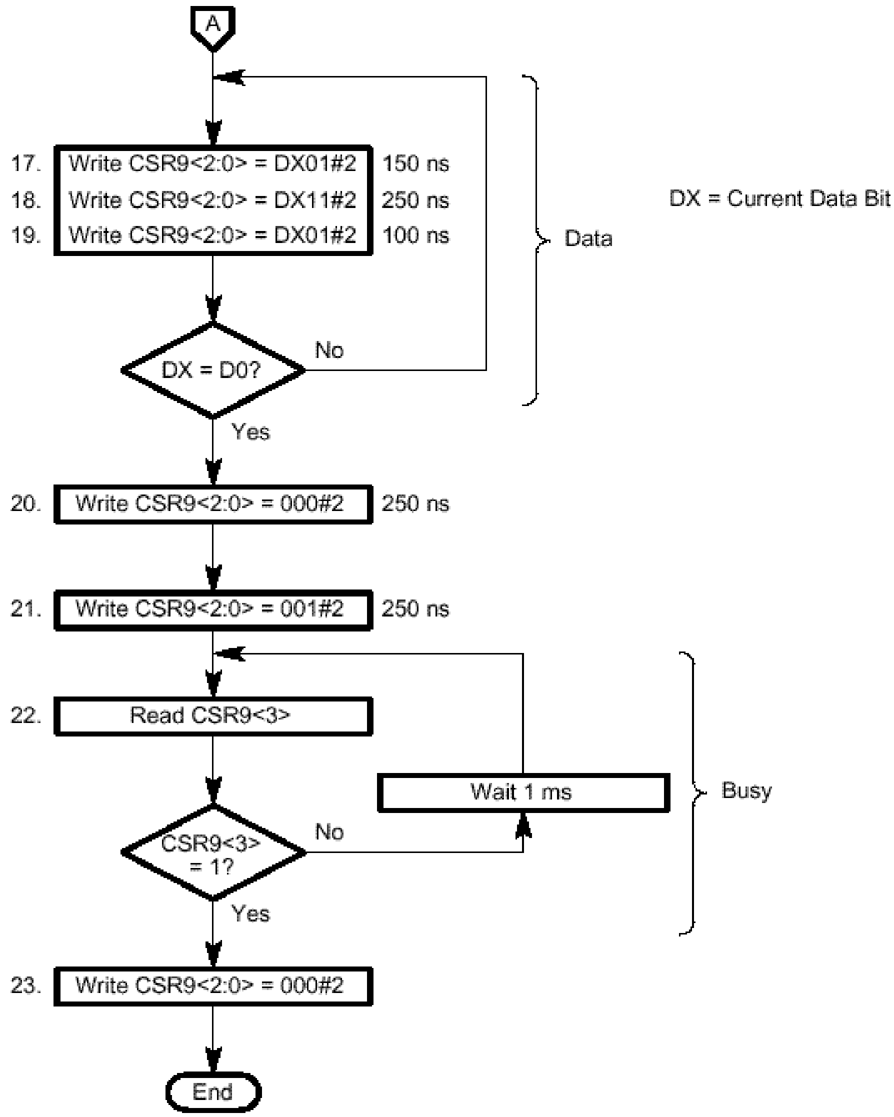

Figure 5. Write Cycle (2 of 2)

The write operation timing of the address and data are showed below:

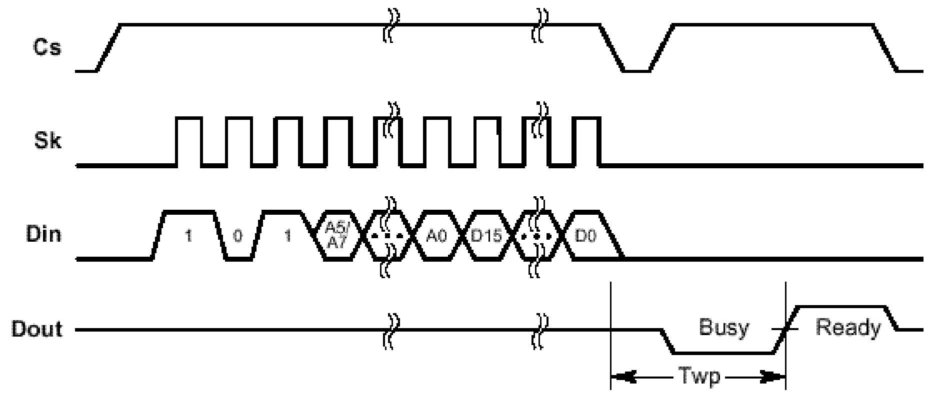

Figure 6. Write Operation

4.0 EEPROM_CRC Calculation Algorithm

The source code for the algorithm used for calculating the CRC is as follows:

```
unsigned short CalcEEPromCrc(unsigned char *EEPromData);
#define DATA_LEN 126 // 1024 bits EEPROM
struct {
        unsigned char EEPromData[DATA_LEN];
        unsigned short EEPromCRC;
        } EEProm;

main() {
          EEProm.EEPromCRC = CalcEEPromCrc(&EEProm.EEPromData);
         }

unsigned short CalcEEPromCrc(unsigned char *EEPromData)
{
#define POLY 0x04C11DB6L
  unsigned long crc = 0xFFFFFFFF;
  unsigned long FlippedCRC = 0;
  unsigned char CurrentByte;
  unsigned Index;
  unsigned Bit;
  unsigned Msb;
  int i;
```

```
 for (Index = 0; Index < DATA_LEN; Index++)
{
  CurrentByte = EEPromData[Index];
  for (Bit = 0; Bit < 8; Bit++)
  {
     Msb = (crc >> 31) & 1;
     crc <<= 1;
     if (Msb ^ (CurrentByte & 1))
     {
        crc ^= POLY;
        crc |= 0x00000001;
     }
     CurrentByte >>= 1;
  }
}
for (i = 0; i < 32; i++)
{
 FlippedCRC <<= 1;
 Bit = crc & 1;
 crc >>= 1;
 FlippedCRC += Bit;
}
crc = FlippedCRC ^ 0xFFFFFFFF;
return (crc & 0xFFFF);
}
```

5.0 Sample Contents of EEPROM

Following is the sample template of the contents of the EEPROM:

74 27 02 00 00 00 00 00 00 00 00 00 00 00 00 00 FF 00 00 04 00 00 00 00 00 00 00 00 00
00 00 01 74 27 4A 10 16 12 13 11 FF FF 00 00 00 00 0C 80 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00