# Cambridge CMOS Sensors

**is now**

# Member of the ams Group

The technical content of this Cambridge CMOS Sensors (CCS) document is still valid.


**Contact information:**

**Headquarters:**
ams AG
Tobelbader Strasse 30
8141 Premstaetten, Austria
Tel: +43 (0) 3136 500 0
e-Mail: ams_sales@ams.com


Please visit our website at www.ams.com

# CCS811 Application Firmware Download Guide

This application note details the method of reprogramming the CCS811 device with a new Application code binary file over the $I^2C$ interface

It assumes that the $I^2C$ communication protocol, as described in application note CC-000803-AN has been implemented (including control of the nWAKE signal).

### *Deciding to download a new Application code binary file*

If a new Application code binary file is available the decision to upgrade may be made by:

1. Reviewing the changes documented in the appropriate release note
2. Checking there are no restrictions associated with the new Application code binary file, such as HW version or Bootloader Firmware version.

If not known the CCS811 Hardware version or Bootloader Firmware version should be checked before attempting to download a new Application code binary file

- The HW Version is stored at register 0x21.
- The FW Boot Version is stored at Register 0x23.

### *Bootloader Mode*

The CCS811 must be in Bootloader mode to download a new Application code binary file.

When a CCS811 device is reset it automatically starts in Bootloader Mode. A reset could be initiated by:

- Powering on the CCS811
- Asserting pin nRESET low for at least 20us and then reasserting high
- Writing the command for a Software Reset

Validating that the CCS811 is in Bootloader Mode can be done by reading bit 7 of the STATUS Register (0x00). In Bootloader mode bit 7 is '0', in Application Mode bit 7 is '1'.

### *Software Reset*

To assert SW_RESET a sequence of four bytes must be written to register 0xFF in a single $I^2C$ sequence: 0x11, 0xE5, 0x72, 0x8A.

### *Firmware Download*

The flow of commands and actions required for a CCS811 Application code binary file download is illustrated below:

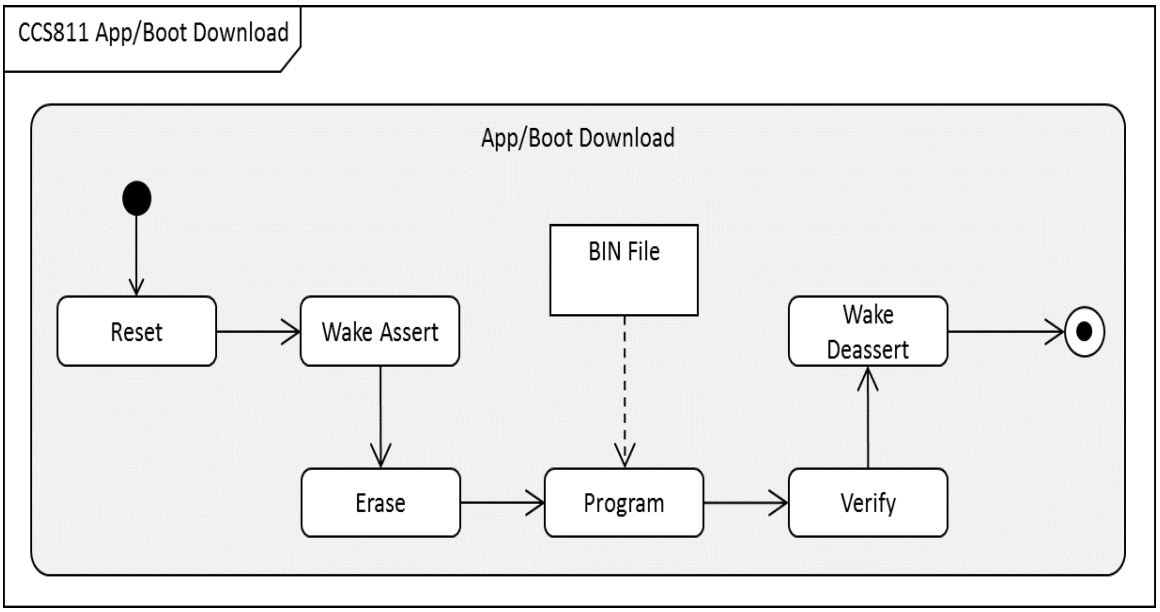


**Figure 1: Application/Boot Code Download Activity Diagram**

The BIN file contains the CCS811 application code binary. This is in a proprietary format and the contents cannot be modified by the user.

### *Erase*

To erase the CCS811 application code, send the following $I^2C$ byte sequences to the ERASE Register (0xF1) of the CCS811: 0xE7, 0xA7, 0xE6, 0x09.

This four byte sequence is required (to prevent accidental erase operations).

The APP_VALID bit [4] of the STATUS Register (0x00) will be cleared by the CCS811 after issuing and successfully processing the above command. After issuing the ERASE command the application software must wait 300ms before issuing any transactions to the CCS811 over the I2C interface.

### *Program*

The following diagram illustrates the process of writing bytes extracted from a BIN file to the CCS811:

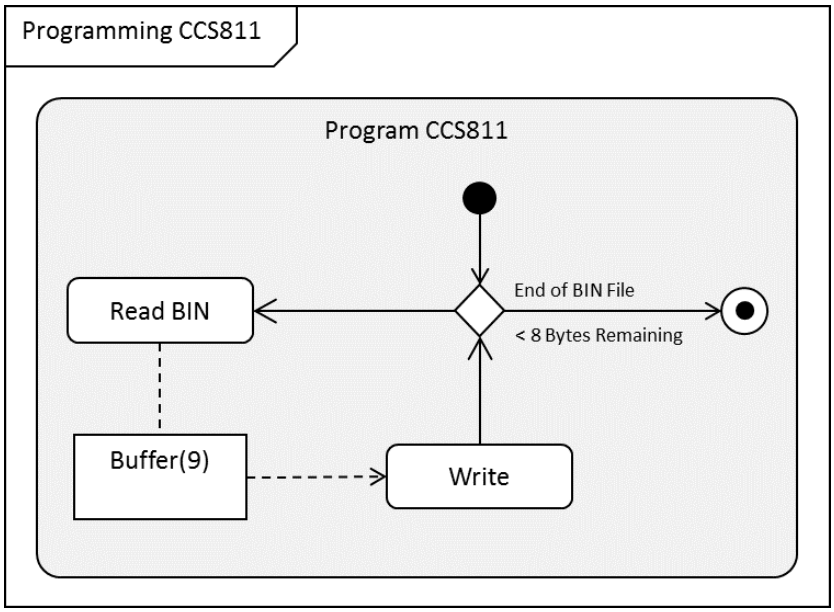


**Figure 2: CCS811 Programming Activity Diagram**

The application binary code length is in multiple of 8 bytes. For every 8 bytes of program code read from the BIN file, the following 9 bytes $I^2C$ data payload is constructed, where REG_BOOT_APP is the command to Write Data to FLASH (0xF2):

LSB MSB

| REG_BOOT_APP (1 byte) | PROGRAM BIN (8 bytes) |
|---|---|

**Figure 3: CCS811 Program Write Payload**

This $I^2C$ payload is then sent to the CCS811 as a single sequence (effectively writing the 8 bytes to Register REG_BOOT_APP (0xF2).

The process is repeated until all program code has been read from the source BIN file and sent to the CCS811.

### *Verify*

To verify the program code has been received error free by the CCS811 device, the host can send the VERIFY command by doing a single byte write of 0xF3.

The APP_VALID bit [4] and APP_VERIFY bit [5] of the STATUS Register (0x00) will be set by the CCS811 if the VERIFY operation was successful. After issuing the VERIFY command the application software must wait 70ms before issuing any transactions to the CCS811 over the I2C interface.

### *Pseudo-Code Example*

The following pseudo-code example code illustrates the overall programming flow for performing an application code binary file download on CCS811:

```
const byte REG_BOOT_APP = 0xF2;
const byte[] CCS811_REG_STATUS = { 0x00 };
const byte[] CCS811_ERASE = { 0xF1, 0xE7, 0xA7, 0xE6, 0x09 }
const byte[] CCS811_VERIFY = { 0xF3 };
byte buffer[] = new byte[61];

// Pulse Reset Pin
WriteLatch(device, 0x00, CCS811_RESET_PIN); Sleep(100);
WriteLatch(device, 0xFF, CCS811_RESET_PIN); Sleep(100);

// Set WAKE low
WriteLatch(device, 0x00, CCS811_WAKE_PIN); Sleep(100);

// Erase application
writeBus(ref device, CCS811_ERASE, 5); Sleep(500);

// Read file in 8-byte chunks until the end is reached, and send each to the board
uint length = ProgramBinFile.Length;
byte *s = ProgramBinFile.ReadBytes();
byte payload[] = new byte[9];
payload[0] = REG_BOOT_APP;

for (int i = 0; i < length; i+=8){
    for (int j = 0; j < 8; j++)
    {
        payload[j + 1] = *(s + (i * 8) + j);
    }
    writeBus(device, payload, 9); Sleep(50);
}

// Verify application
writeBus(device, CCS811_VERIFY, 1);Sleep(500);
writeBus(ref device, CCS811_REG_STATUS, 1);
readBus(device, ref buffer, 1);
if ((buffer[0] & 0x30) == 0x30){
    // program code valid
}
else{
    // program code invalid
}
```

# References

| Document Reference | Description |
|---|---|
| CC-000619-DS | Datasheet for CCS811 |
| CC-000803-AN | CCS811 Programming and Interfacing Guide |