1 Background

CambridgeIC has a range of Integrated Circuits for processing resonant inductive position sensors. Central Tracking Units (CTUs) include the CAM312, CAM502 and CAM204. Resonant Inductive Encoder ICs include the CAM622. For the purpose of this document they will all be referred to as CTUs.

Software inside CTUs is partitioned into two fields: Application Code and Bootloader Code. Application Code is responsible for normal operation including taking measurements and interfacing the results to a host device. The Bootloader Code allows a host device to update the Application Code using the CTU's SPI interface. In normal operation, the version number of the Application Code (the System Version Number) can be read over the SPI interface from the SYSVER register. The version number of the Bootloader Code can be read from the BOOTVER register.

1.1 Reasons to Update Application Code

Updating a CTU's Application Code allows a customer to:

- Use new features during product development, for example:
  - A new interface
  - Support for a new peripheral
  - Support for a new types of sensor
- Fix an issue with the CTU's old Application Code
  - While the CTU is in development at CambridgeIC
  - Theoretically in the field, although this has never been needed
- Use older Application Code that a customer used for their product qualification, in cases where the production version includes newer Application Code whose new features are not required.
- Use a CTU whose production version does not include Application Code, such as the CAM312ME-0002

1.2 Methods of Updating Application Code

CTUs include an SPI interface, and this is always used for updating Application Code.

Many applications for CTUs also use SPI as a primary communications interface, for communicating measurement results. If a customer's product includes a CTU connected to a host device such as a microcontroller on the same PCB then it is possible for a customer to update Application Code through this host. To do this, a customer needs to implement the programming protocols detailed in section 4. Please contact CambridgeIC for assistance, including sample code.

For other applications it may be more convenient to program the CTU independently, by connecting a programming device to its SPI lines. CambridgeIC has two suitable programming devices, both of which connect to a PC running software available from CambridgeIC. The older CTU Adapter and its software is described in section 3.1, and the newer Streaming Adapter and its software is described in section 3.2. This method is usually easier during early prototyping, or when the CTU is mounted to a PCB that does not include a host.

When using an Adapter for programming, the Adapter acts as an SPI controller. If it is connected to a CTU that is also connected to a customer's host then it is necessary to either physically disconnect the host from the SPI lines (and, if used, nRST), or to put the host's SPI outputs into a high impedance state.

When developing a PCB that includes a CTU, it is recommended to make allowance for connecting an Adapter to the SPI lines (ideally plus nRST). Where space is at a premium test pads are adequate. Where space and cost are less of an issue adding a connector footprint to the PCB design is recommended.

2 CTU Firmware File Format

New CTU Application Code is provided in the CTU Firmware File format (.cff). This comprises a Header Block, a Data Block and a Checksum Word. The Header Block is plain text (each byte representing a text character using ASCII codes), while the Data Block and Checksum Word are binary.

The Header Block can be viewed on a PC as a text file. The text includes version number, build date and a description of the CRC algorithm that can be used to checksum the Data Block.

The Header Block comprises rows that start with the "#" character and end in line feed ("<LF>"). The data block can be identified by searching for the character after <LF> that is not "#". In byte representation, this means the bytes that follow the first occurrence of 0x0A that is not immediately followed by 0x23.

The Data Block comprises the remaining bytes in the .cff file. The last two bytes are the Checksum Word for the Data Block. This may be used to verify that the Data Block is valid and uncorrupted. If the Data Block has an odd number of bytes, pad with 0x00 at the end so that it includes only complete words.

3 Programming Using an Adapter and PC

Adapters connect to a PC over USB and to a CTU over SPI. They allow programming software running on a PC to communicate with CTUs, including updating Application Code.

If the CTU to be programmed is on a Development Board from CambridgeIC then connect it directly to the Adapter using a 14-way SPI connecting cable.

If the CTU is mounted on a customer's PCB with then connect the signals shown in Table 1 using appropriate custom wiring.

If the host device applies power to the CTU then DO NOT connect the Adapter's supply output to it (VSUPPLY or VCTU)

If the host device does not supply power to the CTU then the Adapter can power the CTU circuitry instead, providing the supply current it draws (including any other customer circuitry that is also connected) does not exceed the maximum power supply output current for the Adapter.

3.1 CTU Adapter

Figure 1 CTU Adapter

The CTU Adapter is illustrated in Figure 1. Please refer to its datasheet for full details. Application Code may be programmed using the following software:

- Update CTU Firmware, which is one of the applications included in CambridgeIC CTU Software.
- updateCTU.exe, which is a program that can be run from the command line for ease of integration with a customer's production test software

These are available as downloads from CambridgeIC's web site's resources page, once a user is registered and logged in.

The CTU Adapter's VCTU supply voltage output may be switched on and off under software control. When performing a "Reset Entry Method" update of Application Code (section 4.2), the software above performs a reset in two ways. It toggles VCTU off and on prior to programming, and also toggles nRST low. This means that the CTU will be reset either if it is powered by the CTU Adapter or if nRST is connected, or both.

If the CTU Adapter does not supply power to the CTU and nRST is not connected then the user must ensure that the CTU is reset prior to an update of Application Code, and that nSS then remains high until the CTU Adapter pulls it low.

3.2 Streaming Adapter

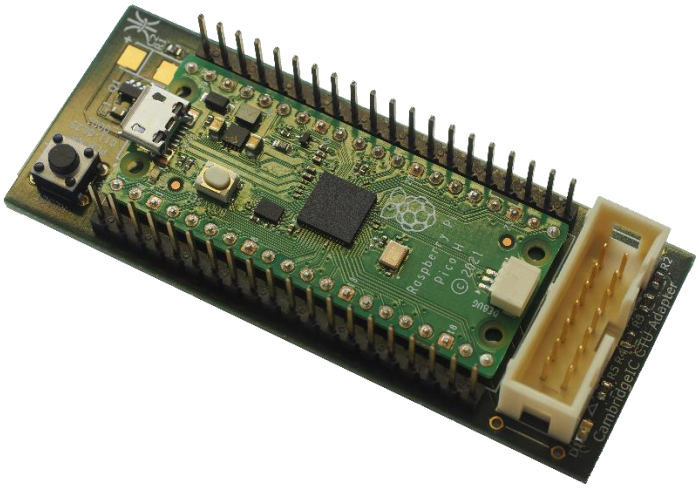

Figure 2 CTU Streaming Adapter

The Streaming Adapter is illustrated in Figure 2. Please refer to its datasheet for full details.

Application Code may be programmed using "Firmware update" software. This may be run as a stand-alone Windows application. Alternatively, it may also be run from a command prompt for ease of integration with a customer's production test software. Please contact CambridgeIC to obtain this software.

The Streaming Adapter has a supply voltage output VSUPPLY which is always on when the device is plugged into the PC's USB port. Unlike the CTU Adapter it is not switched and under software control. This means that the Streaming Adapter can only force a CTU hardware reset if the Streaming Adapter's nRST line is connected to the CTU circuitry's nRST input.

If the Streaming Adapter's nRST line is not connected to the CTU circuit then the user must ensure that the CTU is reset prior to an update of Application Code, and that nSS (nCS) then remains high until the Streaming Adapter pulls it low.

4 Programming using a Host Device

As noted in section 1.2, the CTU usually communicates with a host device over SPI. Typically the host device is a microcontroller including a customer's embedded software. If the host and CTU are connected together in a customer's product then it may be convenient for the customer to implement the bootloader process inside their host.

4.1 Programming Signal Descriptions

Table 1 lists the connections to the CTU circuitry that are used for programming. SPI signal names depend on the CTU that is to be programmed, with alternative naming given in brackets in Table 1 and below.

Table 1 Signals required for programming

Signal Name (Alternative)	Description
VCTU (VS)	CTU supply voltage Powered either by the host device or by the Adapter, but never both
nRST	Resets CTU, active low Host connection not essential but recommended
nSS (nCS)	Chip select, active low, input to CTU
SCK	Serial clock, input to CTU
MOSI (SDI)	Serial data output from controller and input to CTU
MISO (SDO)	Serial data output from CTU and input to controller. Requires a pull-up resistor, at least during programming.

Some CTUs include IOs for signalling, and it is not necessary to connect these for programming Application Code.

4.2 Bootloader Process

The process for programming Application Code is illustrated in Figure 5. The first step is to identify whether or not the CTU already has valid Application Code, because this determines which Bootloader entry method to use.

If the CTU does include Application Code already, or unsure, the host must use the "BOOTLOAD Bit Entry Method". The host must first set the CTU's BOOTLOAD bit. This is located in the SYSCW register in CAM312, CAM502 and CAM204 CTUs, accessed using a normal SPI transaction. Following that SPI transaction, the nSS (nCS) line must remain high for at least a time TnRST2nSS (TnRST2nCS) illustrated in Figure 3. Its value is specified in the datasheet of the relevant CTU.

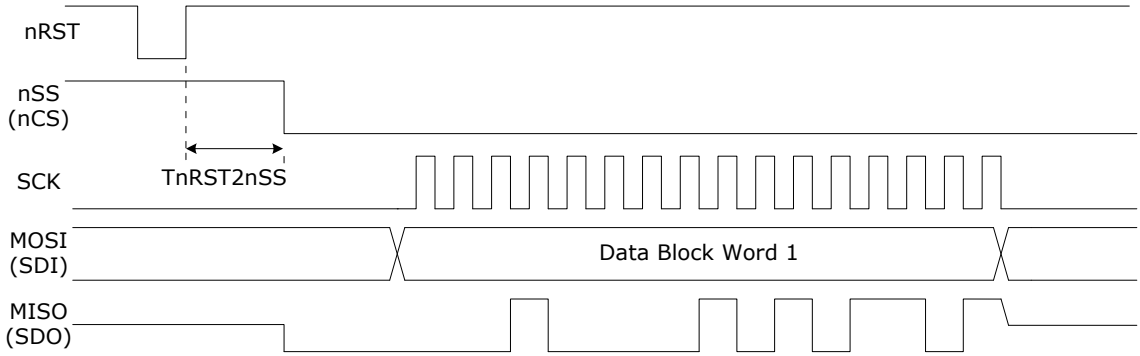

Figure 3 timing of first Data Block Word SPI transaction after setting BOOTLOAD=1

If the CTU does not include Application Code, the host must use the "Reset Entry Method". This begins with the a hardware reset of the CTU. This is done either by pulling nRST low or by removing power (VCTU=0V) then reapplying it. The CTU's nSS (nCS) line must be pulled high during or after reset, and must be pulled low before the first SCK transition. The time between coming out of reset and the nSS (nCS) low edge before the first SCK edge is denoted TnRST2nSS (TnRST2nCS). The minimum value of TnRST2nSS (TnRST2nCS) is specified in the datasheet of the relevant CTU.

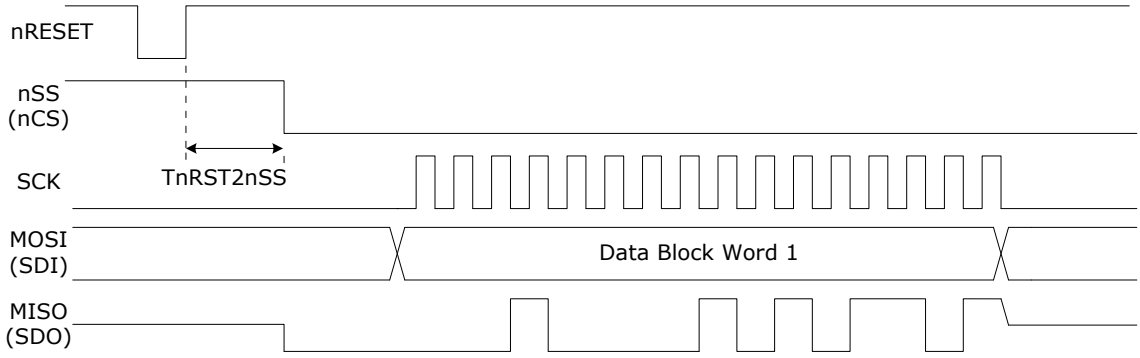

Figure 4 timing of first Data Block Word SPI transaction after reset

The host must now perform the Data Block Transfer Process, as detailed in section 4.3. There are 4 possible results: "successful", "CTU not in Bootloader Mode", "Unexpected CTU Response" or "Check CTU Wiring".

If successful, the host may send the word 0xB002 to the CTU over SPI to request a reset. It must then wait for internal validity checks to complete. Please refer to the CTU's datasheet for details and timings. Once these are complete, the CTU is ready for normal operation running the new Application Code.

If the CTU returns the value of the SYSID register as the first word of the Data Block Transfer Process (factory default 0xABCD) then it is still running old Application Code, and is not in Bootloader Mode. The upload may either be aborted or begun again.

If the Data Block Transfer Process fails with a MISO (SDO) error (equal to 0x0000 or 0xFFFF) then the CTU's wiring should be checked. In particular its power supply, reset circuitry and SPI connections.

If the CTU returns an unexpected response over SPI during the Data Block Transfer Process (typically 0x0BAD) then the Bootloader process has failed and must be restarted. Sending 0xB002 to the CTU over SPI may initiate a reset. Once validity checks have been completed the CTU will return the value of the SYSID register if the host sends 0x0000 over SPI, but only if the CTU still includes old Application Code. If not, the CTU does not include valid Application Code any more, and the Bootloader Process must be repeated.

When implementing the Bootloader process, it is essential to implement both the BOOTLOAD Bit Entry Method and the Reset Entry Method. The Reset Entry Method can not be used when starting with valid Application Code, so the BOOTLOAD Bit Entry Method must always be implemented. However there is always a risk of an unexpected behavior of the host system, for example another device sharing the same SPI device unexpectedly accessing the SPI bus during Bootloader data transfer. This may corrupt CTU Application Code, leaving the part with no valid Application Code. The Reset Entry Method always allows new Application Code to be programmed in this situation.

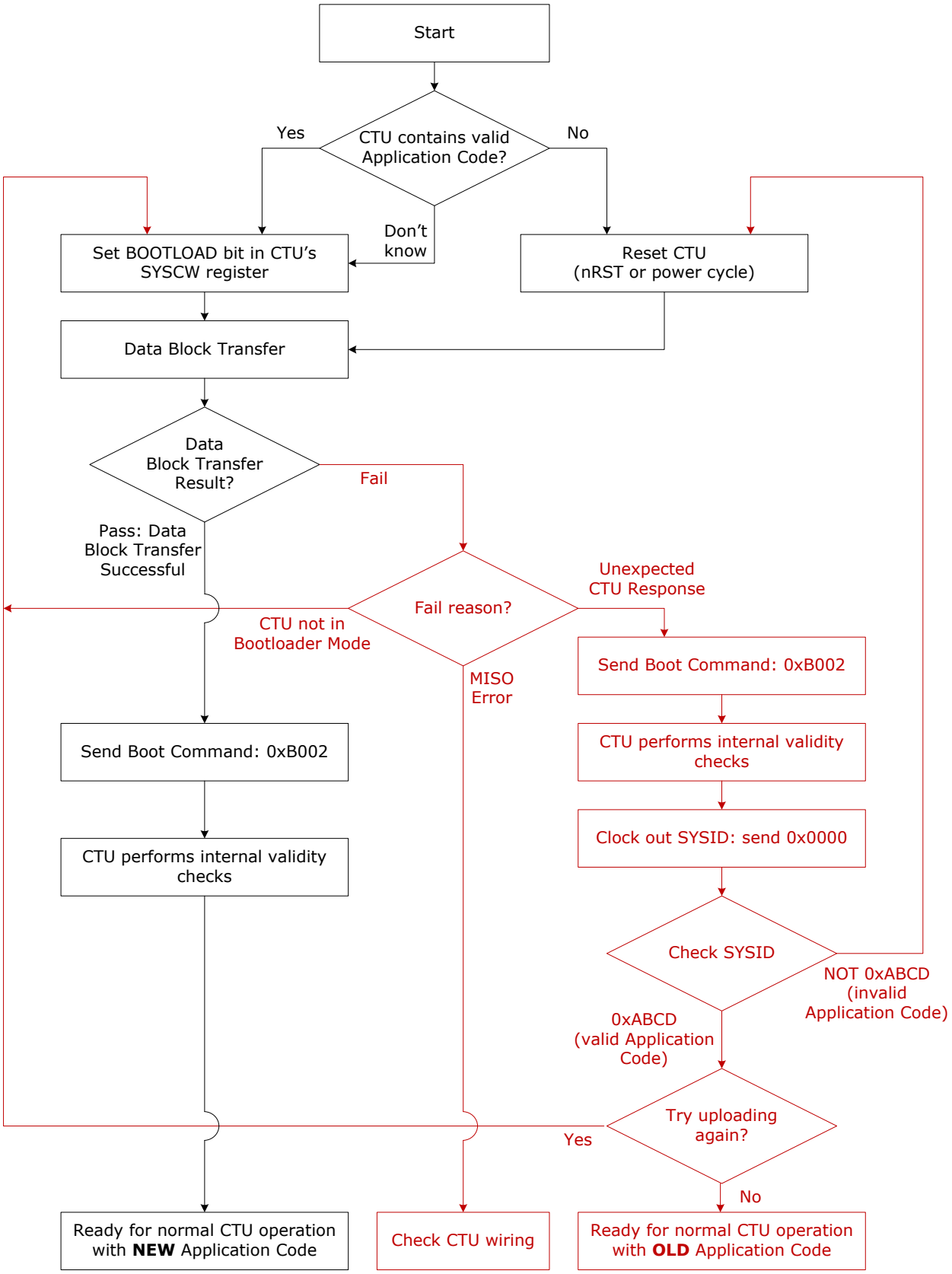

Figure 5 Bootloader flowchart

4.3 Data Block Transfer Process

The Data Block Transfer process is for sending new Application Code to the CTU, and is illustrated in Figure 6. It forms part of the Bootloader Process described in section 4.2.

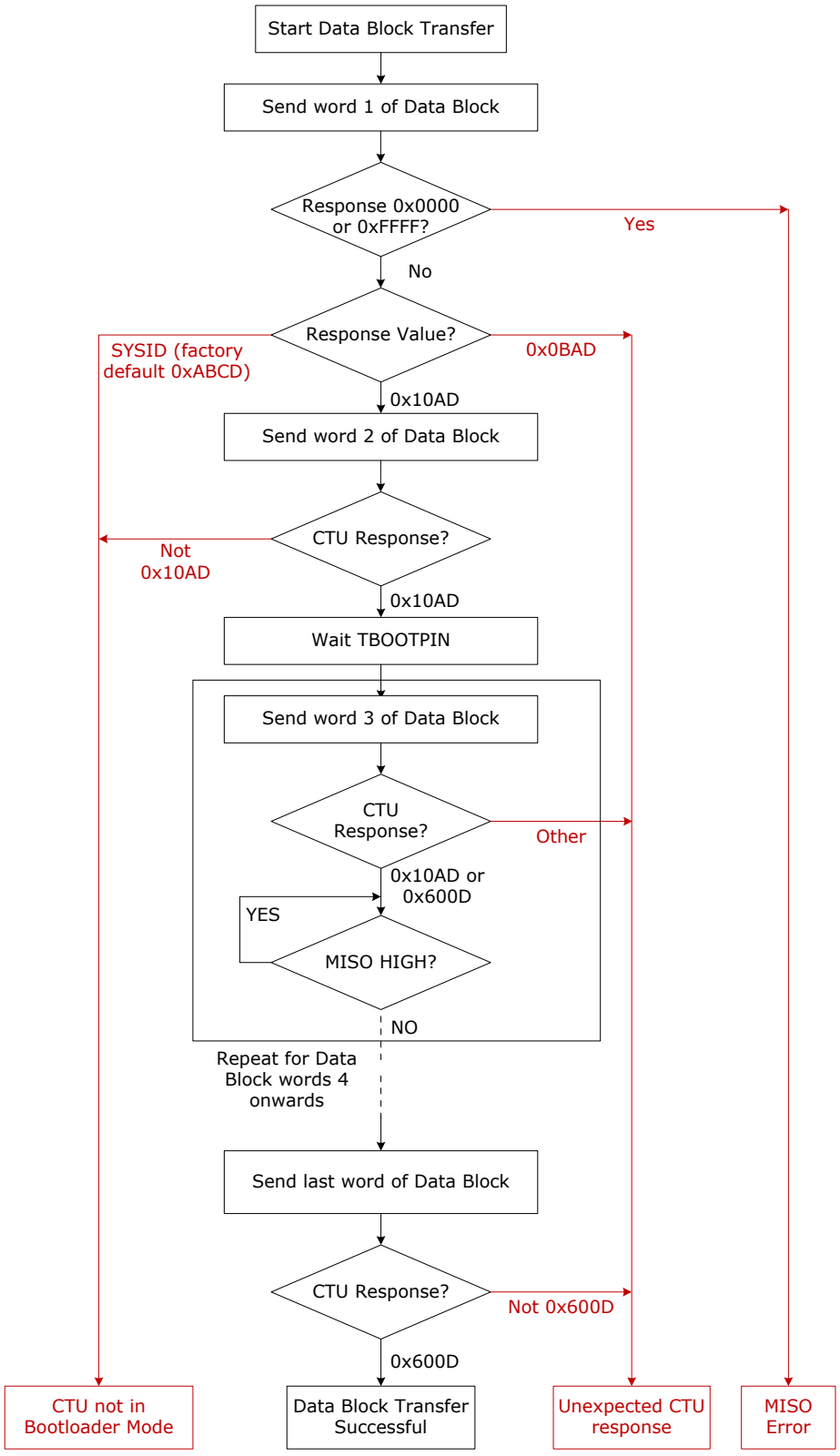

Figure 6 Data Block Transfer Process

Data is transferred in 16-bit words over the SPI interface, as described in section 4.4. Each time the host sends a word, the host should check the response received on the CTU's MISO (SDO) output…

- If the response is 0x0000 or 0xFFFF then there is a MISO (SDO) Error. Possible causes include a wiring fault, the CTU being unpowered, or the CTU being unexpectedly held in reset.
- The response 0x10AD indicates the data transfer process is proceeding successfully.
- The response 0x600D indicates the data transfer process has completed OK.
- If the response is equal to the CTU's SYSID value (0xABCD by default) then the CTU is not in bootloader mode. Data Block Transfer was unsuccessful and should be terminated. The CTU contains valid Application Code. This means that the BOOTLOAD Bit Entry Method should have been successful, and can be retried.
- If the response is 0x0BAD or another response not listed above then the Data Block Transfer process has failed unexpectedly. Possible causes include Data Block data not valid for this CTU, the host sending a data word over SPI too early, SPI interface interference and CTU power supply voltage out of specification.

The host must send the first two words of the Data Block and then pause before the third. The length of the pause is measured from the last SCK edge of the second word and the first SCK edge of the third word, and is denoted TBOOTPIN. A minimum value of TBOOTPIN is required, and it is specified in the CTU's datasheet.

The remainder of the Data Block is then sent to the CTU, one word at a time. The host must check the state of the MISO line after each word. If it is high, the host must pause until MISO has gone low. The time taken for MISO (SDO) to go low is denoted TBOOTWAIT. Its maximum value is specified in the CTU's datasheet.

The CTU's response to the last word of the Data Block will be 0x600D if successful.

4.4 SPI Communication with the CTU in Bootloader Mode

Communication with the CTU in Bootloader Mode uses the same SPI mode and bit ordering as normal operation; please refer to the ship's datasheet. The minimum clock low and high times and the clock period (TSCKL, TSCKH, TSCK) are also the same.

The CTU's SPI select line nSS (nCS) must be low while data is being clocked into the CTU. nSS (nCS) may remain low for the entire Bootloader process. Alternatively, nSS (nCS) may be pulled high after any Data Block Word, providing it is pulled low again before the next.

As noted in section 4.3, the host must check the state of the MISO (SDO) line after sending each Data Block Word. If MISO (SDO) is high, the host must pause communications until the CTU pulls it low. The time between the last SCK low going edge of a Data Block Word and MISO (SDO) going low is denoted TBOOTWAIT. The maximum value is specified in the CTU's datasheet.

Note that the CTU only pulls MISO (SDO) low to indicate it is ready for the next Data Block Word if the nSS (nCS) line is low. If nSS (nCS) is high, MISO (SDO) will be high impedance, to allow other slave devices to share the same SPI bus. If the host chooses to pull nSS (SDO) high after a Data Block Word, it may test the state of MISO (SDO) by pulling nSS (nCS) low again. Providing SCK remains low throughout, this test may be repeated until the host detects MISO (SDO) is low.

If specified for a CTU, a time delay TMISOL2SCKH_BOOT (TSDIL2SCKH_BOOT) is required each bootloader SPI transaction, between the time MISO (SDO) goes low after one SPI transaction and when the host drives SCK high for the next SPI transaction. This is illustrated in Figure 7. The minimum value of TMISOL2SCKH_BOOT (TSDIL2SCKH_BOOT) is specified in the CTU's datasheet.

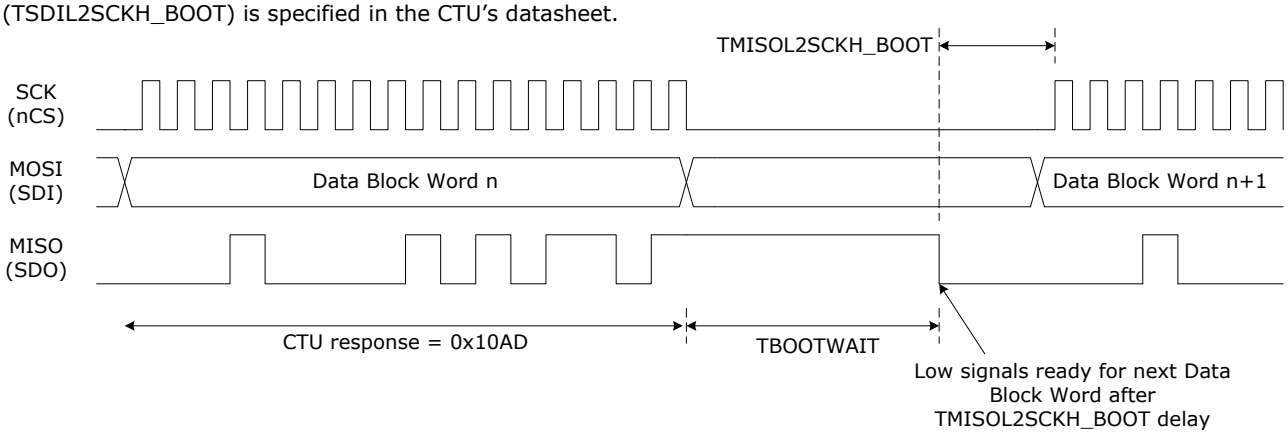

Figure 7 SPI communication, sending Data Block

5 Document History

Table 2 main changes

Rev	Date	Comments
0001	19 July 2023	First draft

6 Contact Information

Cambridge Integrated Circuits Ltd
21 Sedley Taylor Road
Cambridge
CB2 8PW
UK

Tel: +44 (0) 1223 413500
info@cambridgeic.com

7 Legal