1. I2C Device(Slave) Address

Since CubeComputer will not be used as the main OBC of the satellite (therefore it is only the ADCS OBC), it acts as a slave on the system I2C bus with 7-bit addressing with is shown below. The default 8-bit write and read addresses of CubeComputer are also shown below.

Function	I2C 8-bit address in hexadecimal	I2C 8-bit address in binary
Write	0xAE	0b10101110 Write
Read	0XAF	0b10101111 Read

I2C 7-bit address in hexadecimal	I2C 7-bit address in binary
0x57	0b1010111

2. Telecommands(TC) and Telemetries(TLM)

The first byte of a message will determine whether the message is a telecommand or telemetry request + the ID of the telecommand or telemetry request. The MSB determines whether it is a telecommand or telemetry request, and the lower 7 bits contain the ID.

Bit(s)	Data
7(MSB)	0 = telecommand (TC) 1 = telemetry request(TLM)
0:6	Telecommand or telemetry frame ID

The format of the message varies for each telecommand and telemetry ID. The length and content of all the telemetry frames for the ACP are detailed in the tables in ADCS Reference Manual, section 4.5.7, while the telecommands are detailed in ADCS Reference Manual, section 4.5.8.

2.1 I2C Telemetry request format

Telemetry is requested from the ADCS over the system I2C bus by either performing a combined read-write operation (repeated start condition) or a separate master write to select the TLM register, followed by a master read operation.

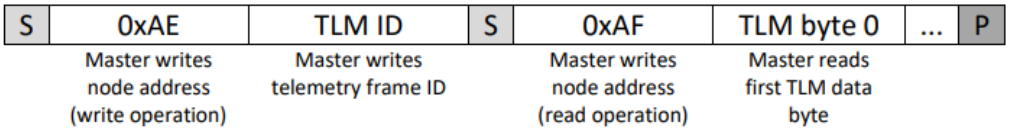


Figure 18: I2C Telemetry request using I2C repeated start condition

Steps to send a TLM:

·Send the start condition.

·Send the node write address (0xAE).

·Send the telemetry frame identifier(mentioned above).

·Send the stop condition.[1]

·Send another start condition.[1]

(Note: In the case of the combined write-then-read operation, the master will issue a repeated start condition (without a preceding stop condition).**)**[1]

·Send the node read address (0xAF).

·The master then issues a number of read cycles depending on the length of the telemetry frame. (Question Mark here)

·Send the stop condition.

(Note: Because the master determines the number of bytes that are read, it is possible to read past the end of a telemetry frame or to read an incomplete telemetry frame. The ADCS will set an error flag if an incorrect number of bytes are read for a given TLM ID. This flag is stored in the Communication Status frame (TLMID 144) and can be read using a telemetry request. The flag will remain set until the Clear Latched Error Flags (TCID 12) is issued.**)**

2.1 I2C Telecommand format

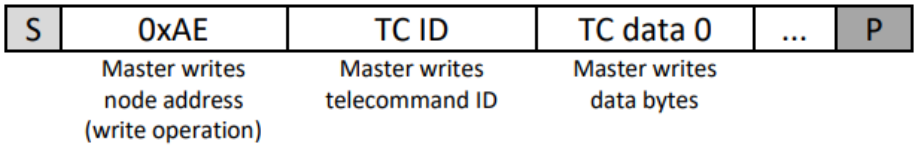


Figure 19: I2C Telecommand

Steps to send a TC:

·Send the start condition.

·Send the node write address (0xAE).

·Send the telecommand identifier.

·Send telecommand parameters.

·Send the stop condition.

(Note: Because the ADCS is an I2C slave, it cannot acknowledge telecommands by performing an I2C write transaction. The telecommand acknowledge status must therefore be polled via a telemetry request (TLMID 240).

It is not a requirement that the telecommand acknowledge status has to be read following a telecommand, but an error will occur if another telecommand is sent before the Telecommand Processed flag (contained in the Telecommand Acknowledge frame) has been set. In this case the telecommand buffer will be overwritten while the first telecommand is being processed, leading to corrupt telecommand data.

The Processed flag is not an indication of the telecommand execution status. The Processed flag is only an indication that the module is ready for another telecommand to be sent.

The Telecommand Acknowledge frame also contains a TC Error flag. This flag will be set if an invalid telecommand ID was received for the last telecommand, or if the number of data bytes were incorrect or contained invalid data.**)**

To ensure proper telecommand execution:

1. Send telecommand.

2. Poll Telecommand Acknowledge Telemetry Format until the Processed flag equals 1.

3. Confirm telecommand validity by checking the TC Error flag of the last read Telecommand Acknowledge Telemetry Format.

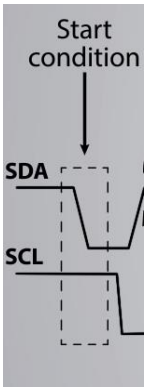
4. Back to step 1 (if another telecommand is to be sent)

(Important note: List of **Telecommands** can be found at **ADCS Reference Manual,** Page 25; List of Telemetry Request can be found at **ADCS Reference Manual,** Page 83)

Example 1: Perform a Reset on the ADCS OBC (Send a telecommand):

ID	1		**Parameters Length (bytes)**		1
Description	Perform a reset				
Parameters	**Offset (bits)**	**Length (bits)**	**Name**	**Data Type**	**Description**
	0	8	Magic number	UINT	Magic number to make sure it is a valid reset command. Should equal 0x5A

Step1: Send the Start condition.(SDA goes low when SCL is high)

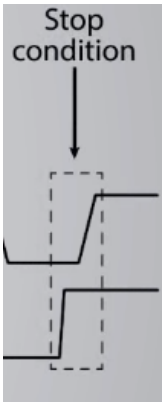


Step2: Send the node write address : 0xAE

Step3: Send the telecommand identifier for reset: 0b0000 0001 → 0x01

Step4: telecommand parameters: 0x5A (Where 0x5A is a magic number to make sure it is a valid reset command.)

Step5: Send the stop condition. (SDA goes high when SCL is high)



Example 2: Check Configuration setting for Unix time flash memory persistence(Send a Telemetry Request):

ID	129		Frame Length (bytes)		2
Description	Configuration settings for unixtime flash memory persistence				
Channels	Offset (bits)	Length (bits)	Name	Data Type	Description
	0	1	Save Now	BOOL	Save current unixtime to flash memory
	1	1	Save On Update	BOOL	Save unixtime to flash memory whenever there is a command to update the unixtime
	2	1	Save Periodic	BOOL	Save unixtime to flash memory periodically
	8	8	Period	UINT	Interval at which to save unixtime to flash memory. (Unit of measure is [s])

Step1: Send the start condition.
Step2: Send the node write address : 0xAE

Step3: Send the telemetry frame identifier: 0b10000001 → 0x81

Step4: Send another start condition.

Step5: Send the node read address : 0xAF

Step6: Send TLM parameters: 0b0010 0000; 0b0011 1100 → 0x20; 0x3C(Save UNIX time to flash memory periodically with the period of 60 seconds)

Step7: Send the stop condition

3. More info about I2C on ADCS-OBC(CubeADCS)

The CubeADCS unit can communicate with other satellite subsystems using the System I2C on the PC104 bus (H1-41, 43). ~~If CubeComputer is the main OBC of the satellite, it will act as a master on the system I 2C bus. Alternatively,~~ it will act as a slave on the system I2C bus ~~if it is only acting as ADCS OBC~~, responding to commands and telemetry requests (as detailed in the CubeADCS User Manual). CubeComputer can be populated with or without pull-up resistors on the system I2C bus.

CubeComputer is set to be a master on the secondary I2C bus (H1-21,23) to communicate with the other Cube Space ADCS modules on the bus. CubeComputer is also connected to the system I 2C bus (H1-41,43). Please indicate the desire configuration for the bus side pull-up resistors on the system I 2C. (Standard option: None)