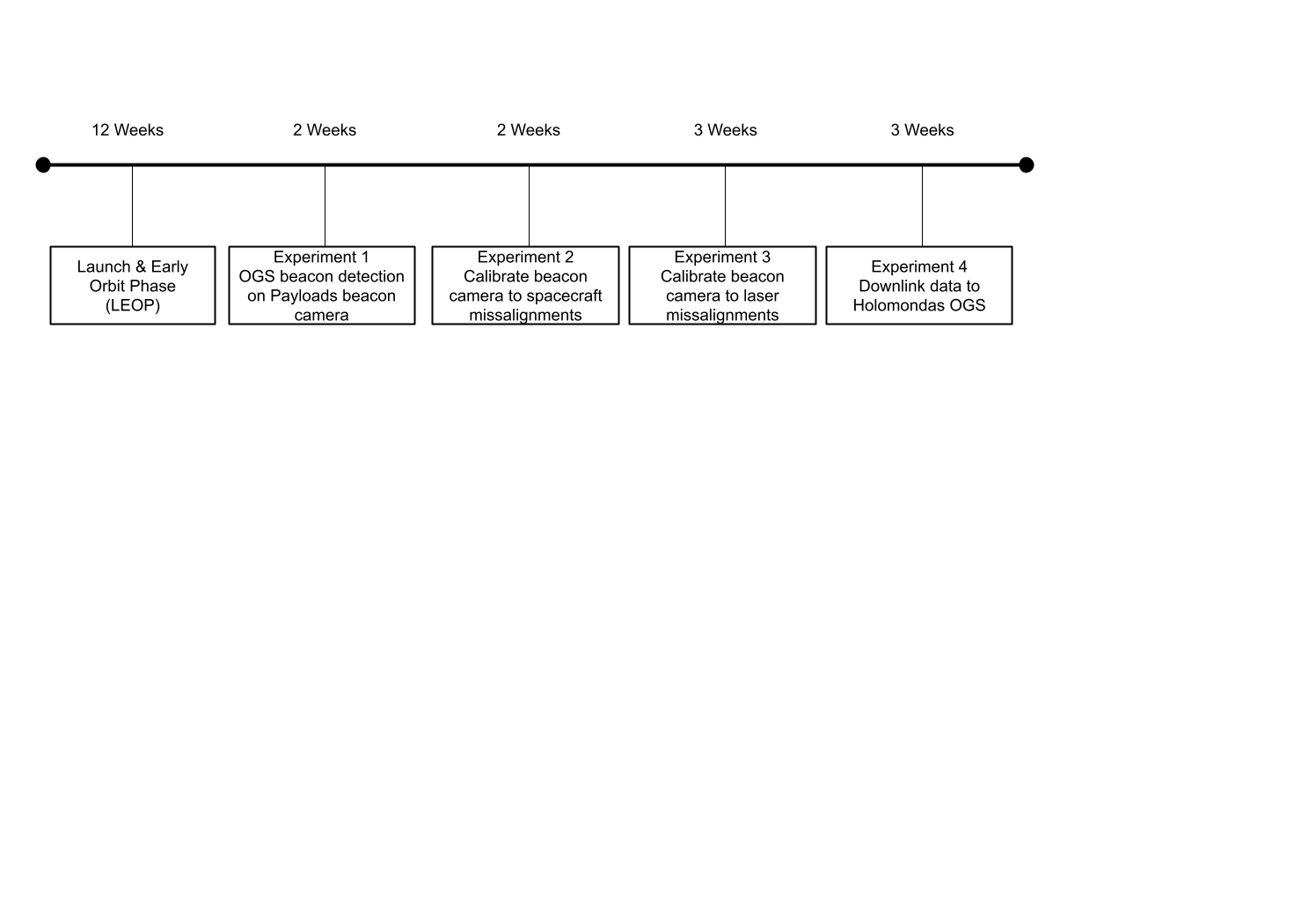
PeakSat – Software

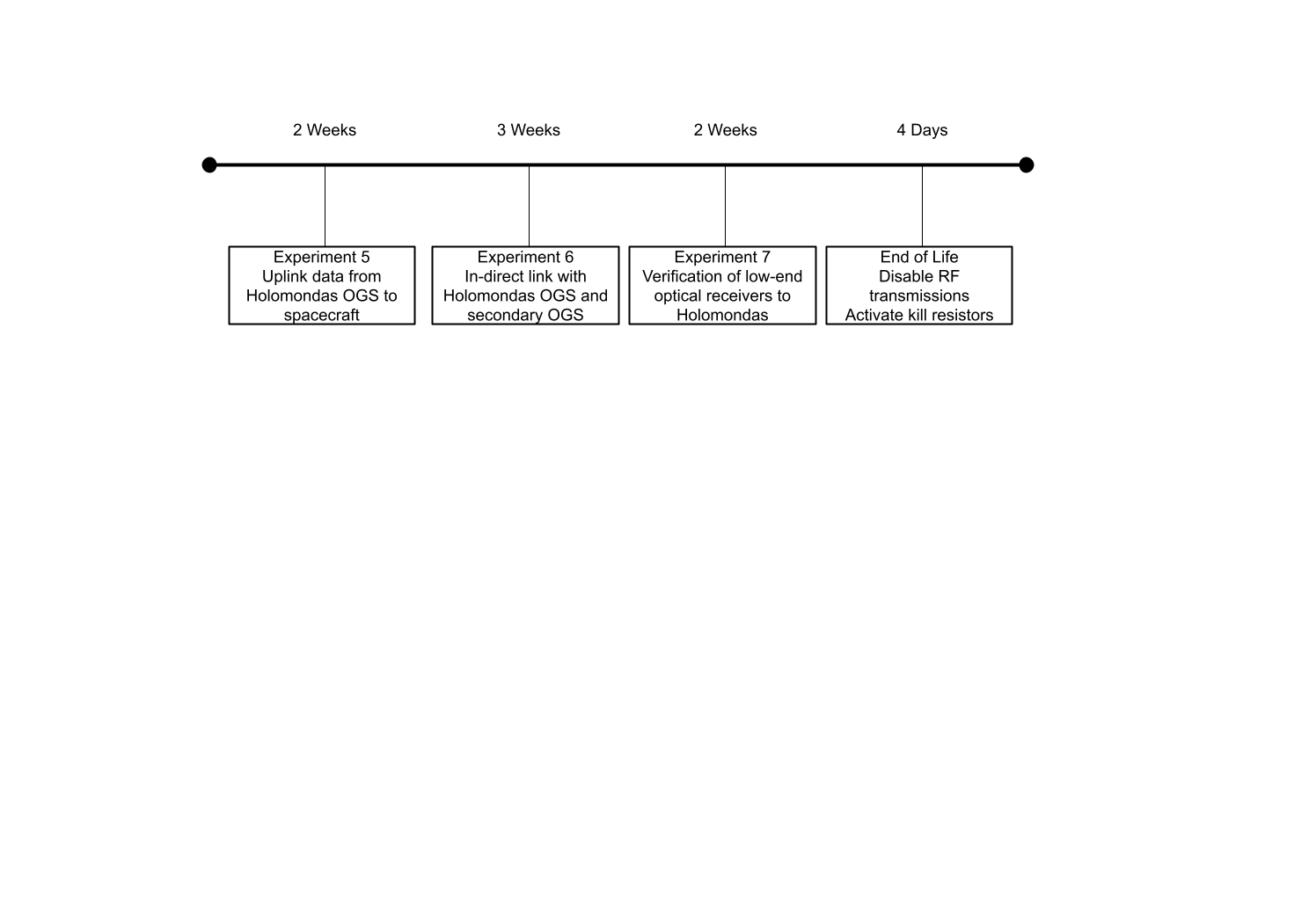
**Mission Design Description – Mission Profile**

**Total mission duration 6 months**

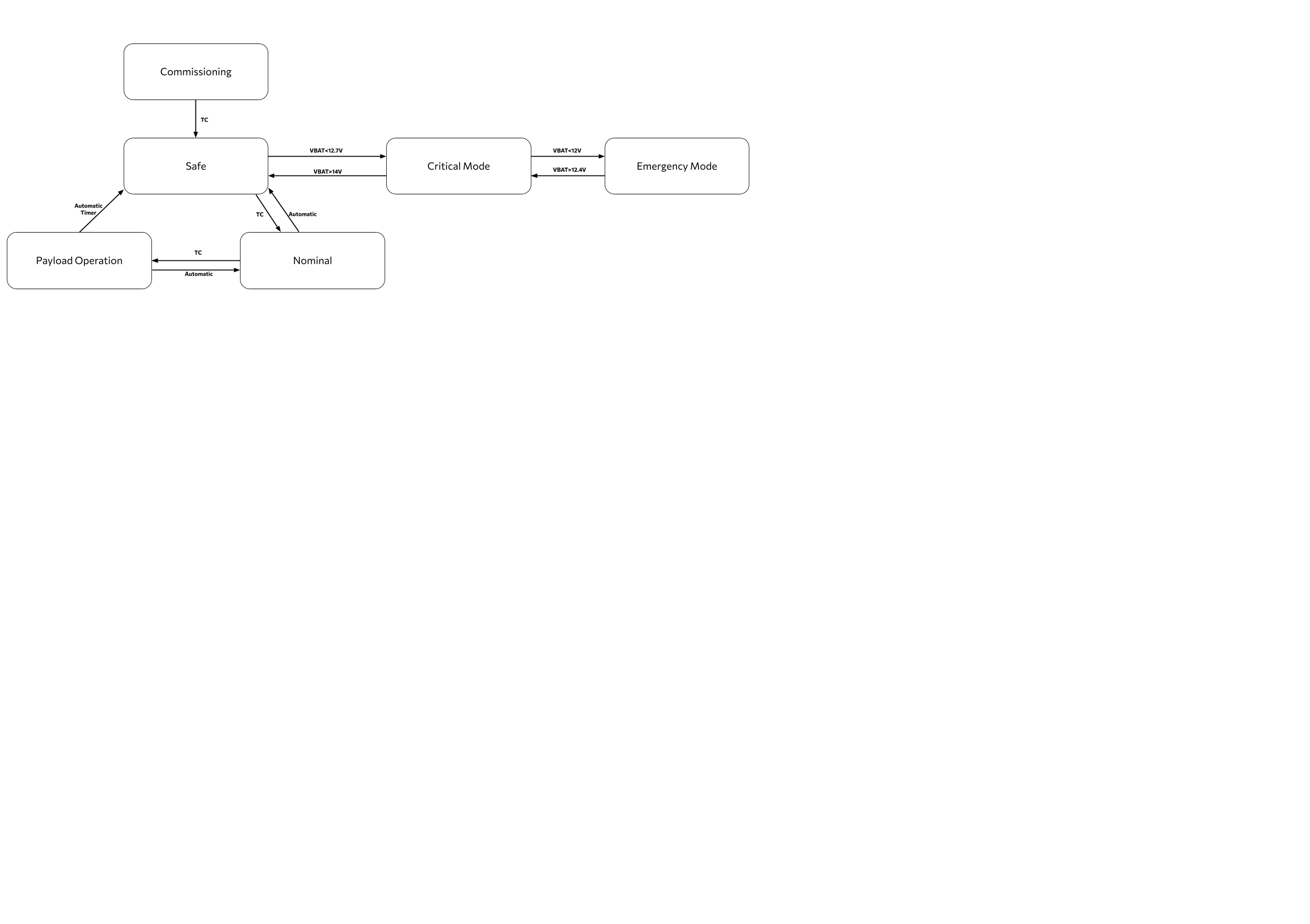
**Primary Mission Objectives**



**Secondary Mission Objectives**



**Operational Modes Overview**



***Critical Mode***

* EPS automatically enters and exits this mode depending on the measured rail voltage.
* Only force-enable output bus channels remain powered
* All output bus channel control commands are rejected in this mode.
* COMMS is forced enable and transmitting only the CW beacon (since the modulated beacon is dependent on OBC while CW is not)

***Emergency Mode***

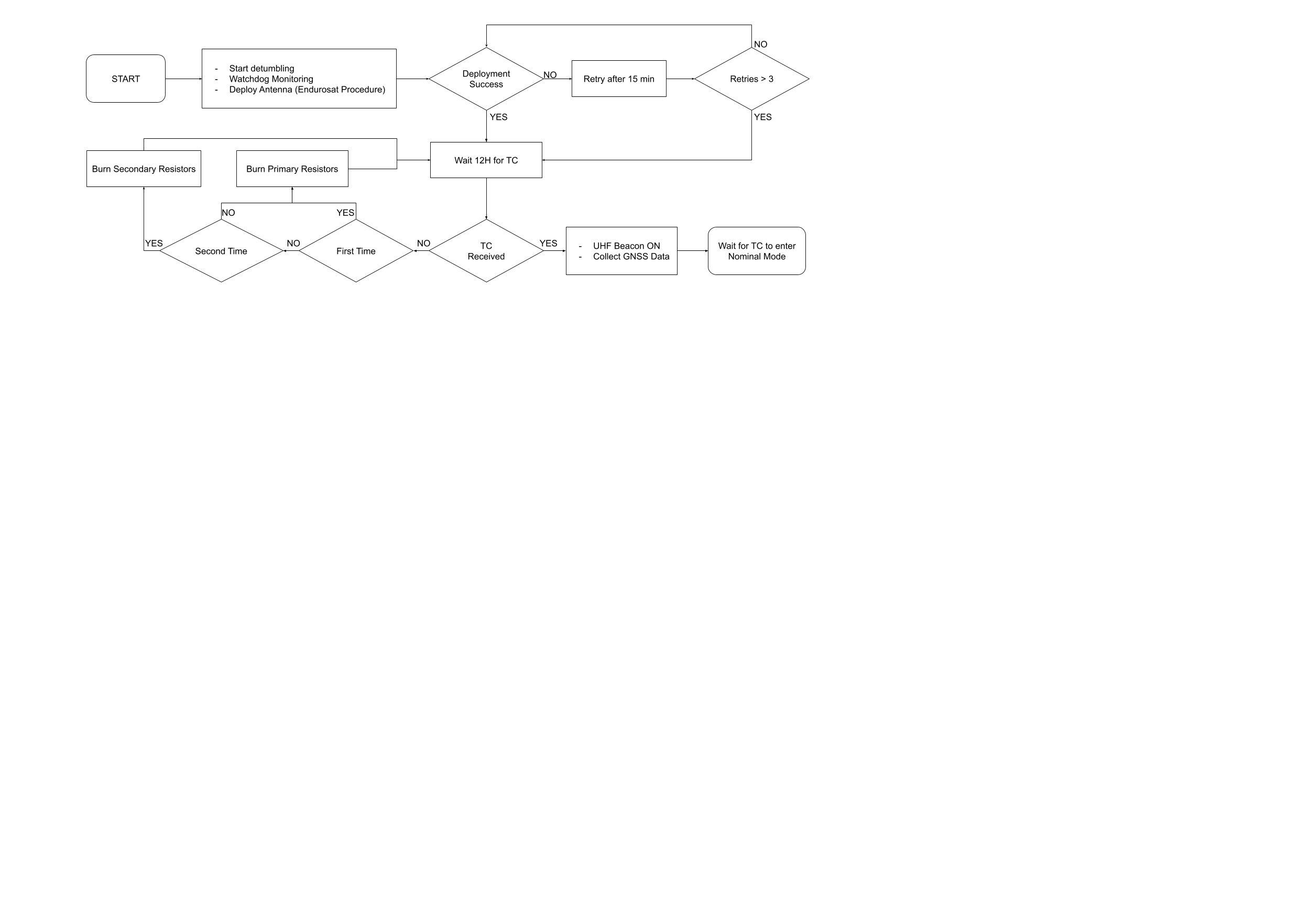
* EPS automatically enters and exits this mode depending on the measured rail voltage.
* All output bus channels are turned off.

*The above modes are hard coded on ISIS EPS. The thresholds can be determined by the team and even configured in-orbit.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Subsystem** | **Commissioning** | **Nominal** | **Safe** | **POM** | **Critical** | **Emergency** |
| **ADCS** | Detumbling Y-Thompson | Sun Pointing | Detumbling  Y-Thompson | OGS tracking | Off | Off |
| **Payload** | Closed | Closed | Closed | Open | Off | Off |
| **COMMS** | UHF beacon Tx, Rx GNSS data | UHF beacon Tx, Rx GNSS data | UHF beacon Tx, Rx  GNSS data | UHF beacon Rx GNSS data | UHF CW beacon RX | Off |
| **OBC** | Open | Open | Open | Open | Off | Off |

***Commissioning Mode***

*Modified Flow chart*



Safe



1. Boot sequence (initialize subsystems and perform health checks).
2. Start 30 min before UHF deployment.
3. If deployment is successful wait for TC for 12 hours. If not reattempt after 15 min. If TC not received after 12 hours reattempt deployment.
4. If deployment not detected after three 15min retries then wait 12h.
5. If TC still not received force primary resistors to burn for 30 sec and wait 12h.
6. If TC still not received force backup resistors to burn for 30 sec and wait 12h.
7. Repeat 5 and 6 until successful deployment.
8. If TC received proceed with collection of GNSS data for 3 days and open UHF beacon.
9. Collect GNSS data on GS and upload the first TLE.
10. Switch to Safe mode via TC from operators.

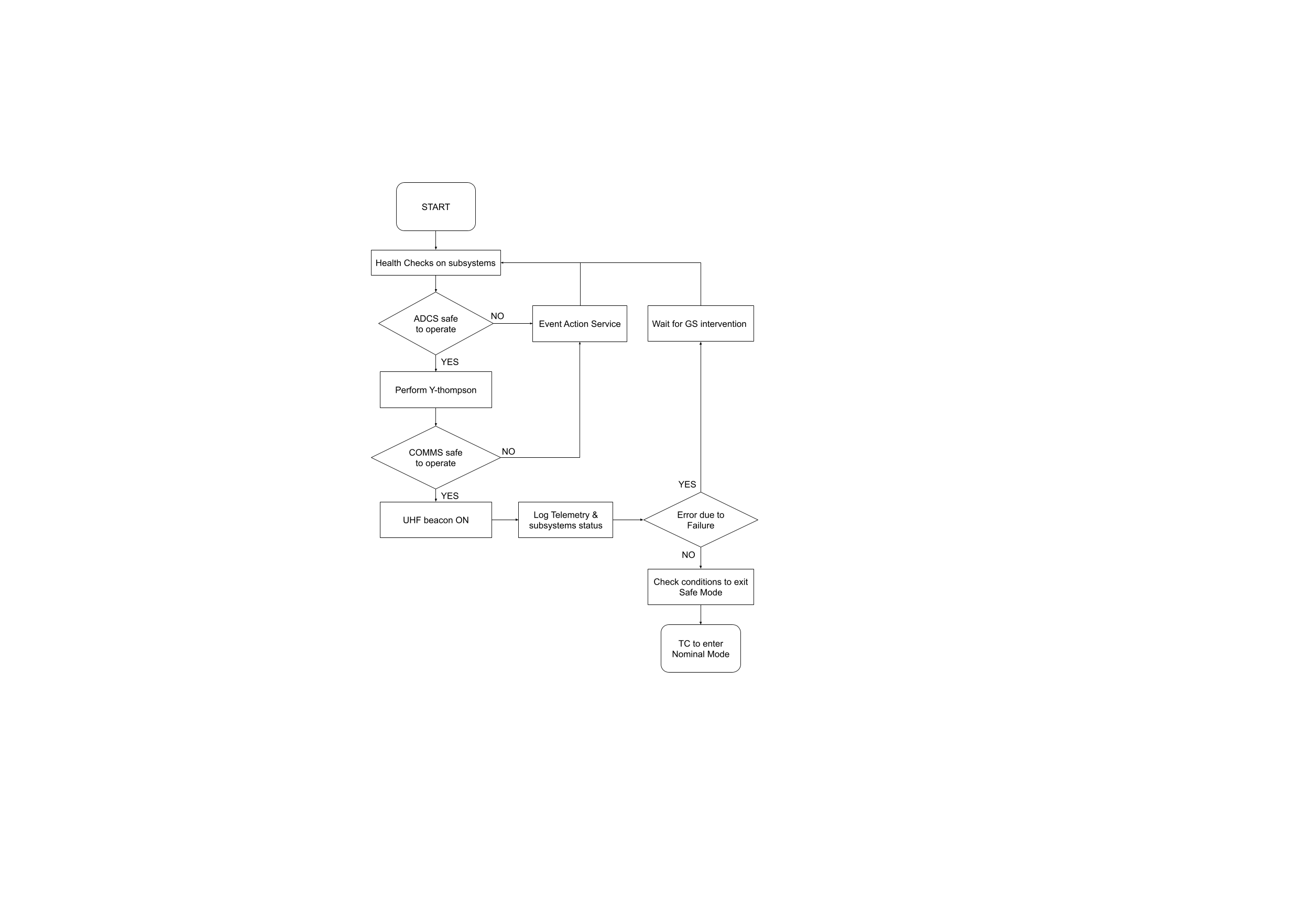
***UHF Deployment Procedure***

Εικόνα που περιέχει διάγραμμα, κείμενο, τεχνικό σχέδιο, Σχέδιο

Περιγραφή που δημιουργήθηκε αυτόματα

* *Endurosat has confirmed the possibility for a force deployment through dedicated GPIO pins.*

***Safe Mode***

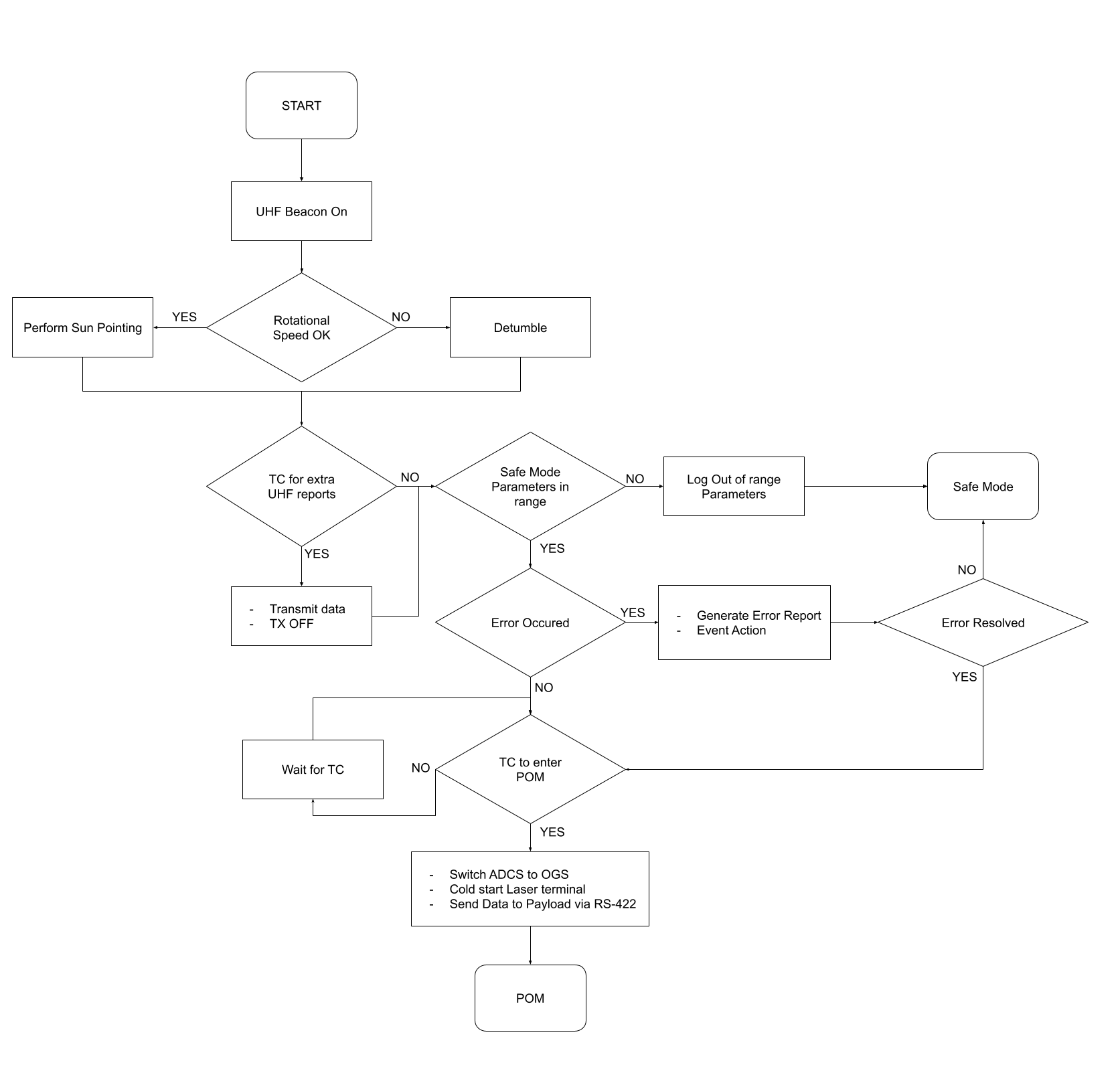


* Payload is OFF.
* UHF beacon ON.
* ADCS in Y-Thompson mode.
* Health checks on subsystems and report to GS.
* GS intervention to fix possible errors.
* Switch to Nominal after TC from operators. If conditions to exit safe mode are not within limits, then SC remains on safe mode.
* All triggers to enter and exit safe mode are configurable in orbit.
* Event actions to be defined during Flatsat integration for subsystem failures. These may include power cycling or complete switching off a subsystem for a given amount of time.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Conditions to enter safe mode (configurable in orbit)** | | | | |
| **Condition ID** | **Parameter** | **Type** | **Units** | **Value** |
| **SM-010** | Battery Voltage | Lower than | V | 13V |
| **SM-020** | Angular Velocity Threshold (1-axis) | Greater than | rad/s | 1.571 (TBC) |
| **SM-030** | CubeADCS CAN bus failure | attempts | - | 5 |
| **SM-040** | COMMS CAN bus failure | attempts | - | 5 |
| **SM-050** | Last contact with GS | Greater than | h | 72 |
| **SM-060** | Multiple resets in given time frame | Greater than | - | TBD |
| **SM-070** | X- Solar panel temperature | Greater than | °C | 80 (TBC) |
| **SM-080** | Y+ Solar panel temperature | Greater than | °C | 80 (TBC) |
| **SM-090** | Y- Solar panel temperature | Greater than | °C | 80 (TBC) |
| **SM-100** | OBC temperature | Greater than | °C | 80 (TBC) |
| **SM-110** | COMMS temperature | Greater than | °C | 80 (TBC) |
| **SM-120** | ADM temperature | Greater than | °C | 80 (TBC) |
| **SM-130** | CubeStar temperature | Greater than | °C | 55 (TBC) |
| **SM-140** | CubeComputer MCU temperature | Greater than | °C | 65 (TBC) |
| **SM-150** | Rate Sensor temperature | Greater than | °C | 55 (TBC) |
| **SM-160** | Magnetometer temperature | Greater than | °C | 75 (TBC) |
| **SM-170** | Battery Pack Temperature | Greater than | °C | 80 (TBC) |
| **SM-180** | EPS board temperature | Greater than | °C | 40 (TBC) |
| **SM-190** | System bus current | Greater than | A | 4 (TBC) |
| **SM-200** | 3V3 channel 1 current | Greater than | A | 1 (TBC) |
| **SM-210** | 3V3 channel 2 current | Greater than | A | 1 (TBC) |

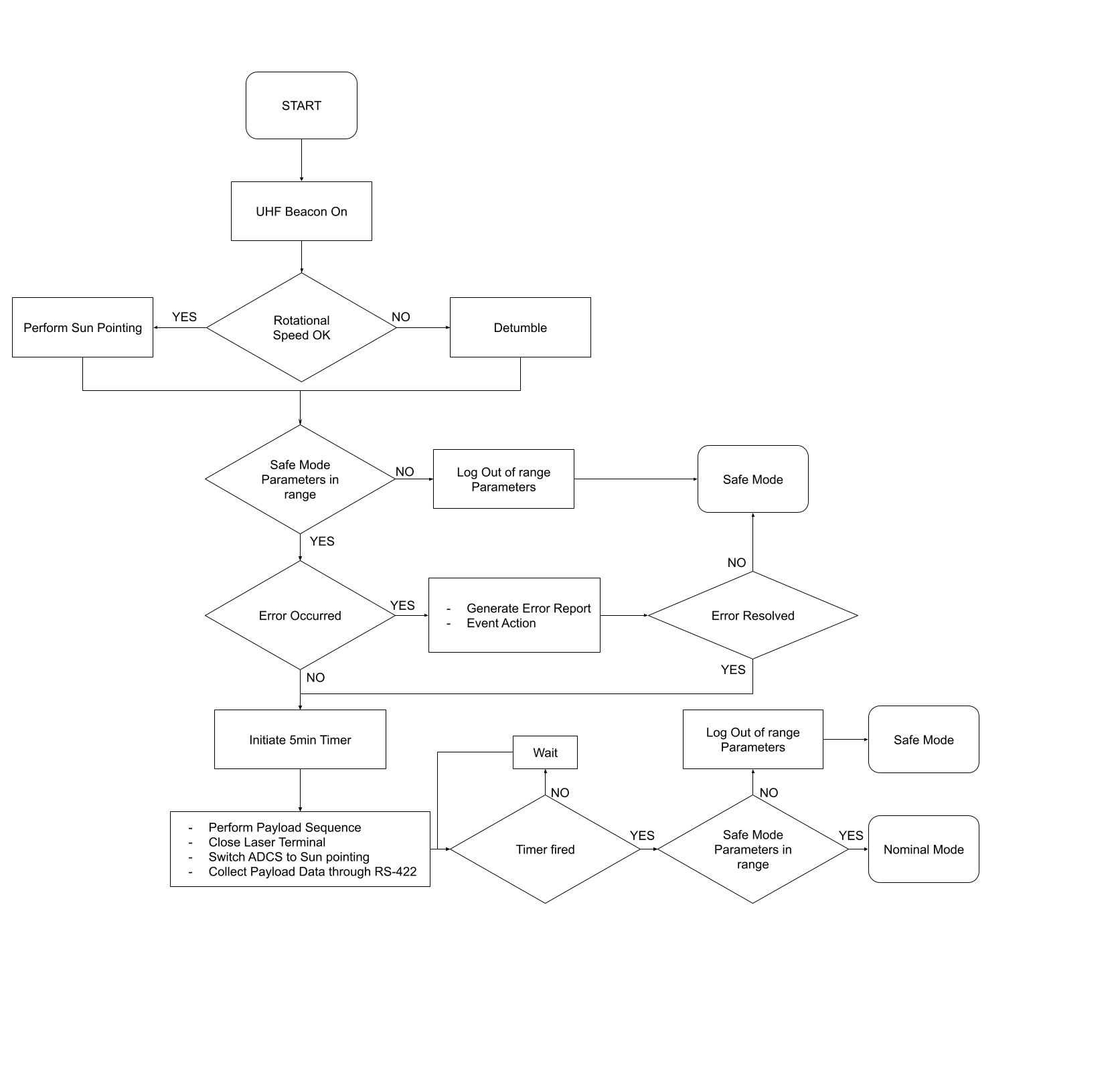
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Conditions to enter safe mode (configurable in orbit)** | | | | |
| **Condition ID** | **Parameter** | **Type** | **Units** | **Value** |
| **SM-220** | 3V3 channel 3 current | Greater than | A | 1 (TBC) |
| **SM-230** | 3V3 channel 4 current | Greater than | A | 1 (TBC) |
| **SM-240** | 5V channel 1 current | Greater than | A | 2 (TBC) |
| **SM-250** | 5V channel 2 current | Greater than | A | 1.5 (TBC) |
| **SM-260** | 5V channel 3 current | Greater than | A | 2 (TBC) |
| **SM-270** | 5V channel 4 current | Greater than | A | 1 (TBC) |
| **SM-280** | 12V channel 1 current | Greater than | A | 2 (TBC) |
| **SM-290** | 12V channel 2 current | Greater than | A | 2 (TBC) |
| **SM-300** | 12V channel 3 current | Greater than | A | 1.5 (TBC) |
| **SM-310** | 12V channel 4 current | Greater than | A | 1.5 (TBC) |
| **SM-320** | 3V3 channel 1 voltage | Out of range | V | 3.15 - 3.45 (TBC) |
| **SM-330** | 3V3 channel 2 voltage | Out of range | V | 3.15 - 3.45 (TBC) |
| **SM-340** | 3V3 channel 3 voltage | Out of range | V | 3.15 - 3.45 (TBC) |
| **SM-350** | 3V3 channel 4 voltage | Out of range | V | 3.15 - 3.45 (TBC) |
| **SM-360** | 5V channel 1 current | Out of range | V | 4.75 - 5.25 (TBC) |
| **SM-370** | 5V channel 2 voltage | Out of range | V | 4.75 - 5.25 (TBC) |
| **SM-380** | 5V channel 3 voltage | Out of range | V | 4.75 - 5.25 (TBC) |
| **SM-390** | 5V channel 4 voltage | Out of range | V | 4.75 - 5.25 (TBC) |
| **SM-400** | 12V channel 1 current | Out of range | V | TBD |
| **SM-410** | 12V channel 2 voltage | Out of range | V | TBD |
| **SM-420** | 12V channel 3 voltage | Out of range | V | 11.5 - 12.5 (TBC) |
| **SM-430** | 12V channel 4 voltage | Out of range | V | 11.5 - 12.5 (TBC) |

***Nominal Mode***



|  |  |
| --- | --- |
| * Payload is OFF, UHF beacon ON. * FDIR from OBC. * ADCS sun pointing (switch to detumbling in case of increased angular velocity). * TC to download data from optical attempts (GNSS, pointing). | * Enter POM after TC with following procedure:   1. Wait for UTC time specified in TC.   2. Initiate timer for POM.   3. Update ADCS TLE.   4. ADCS switch to ground station tracking.   5. Optical terminal cold start (~1min). |

***POM Mode***



|  |  |
| --- | --- |
| * FDIR from OBC * UHF beacon * ADCS ground station tracking * Switch to safe mode via TC * Switch to safe mode in case timer exceeds 5 min | * Exciting POM procedure:   1. Close laser terminal   2. Switch ADCS to sun pointing   3. Collect data on OBC through RS-422   4. Turn off the payload |

***CONOPS***

* 100mbps streaming is not possible from the OBC. **Data will be prestored in NAND of OBC**.
* Payload is set to idle mode 10 minutes prior to actual optical downlink. During these 10 minutes up to 18MB of data can be stored in payload through RS-422 (ADM3483 transceiver is used with default data rate at 250kbps).
* During operations, **two sets of predefined data each 18MB in size**, will be pre-written to the **NAND memory of the On-Board Computer (OBC) before launch**.
* To ensure data integrity, *checksum validation* is performed. Should the checksum verification fail, the data will be re-read to address any potential errors caused by degradation. If the checksum fails a second time, the OBC will proceed to access the redundant set of data.
* In case data transmission is not possible through RS-422 (e.g. in case redundant MCU has taken over which does not have access to NAND) Astrolight provides the ability to store up to **1MB of data in non-volatile memory**. These data will again be **stored on the payload prior to launch** as a fallback measure.
* Continuous repetition of both above data sets is only a minor degradation for the mission.

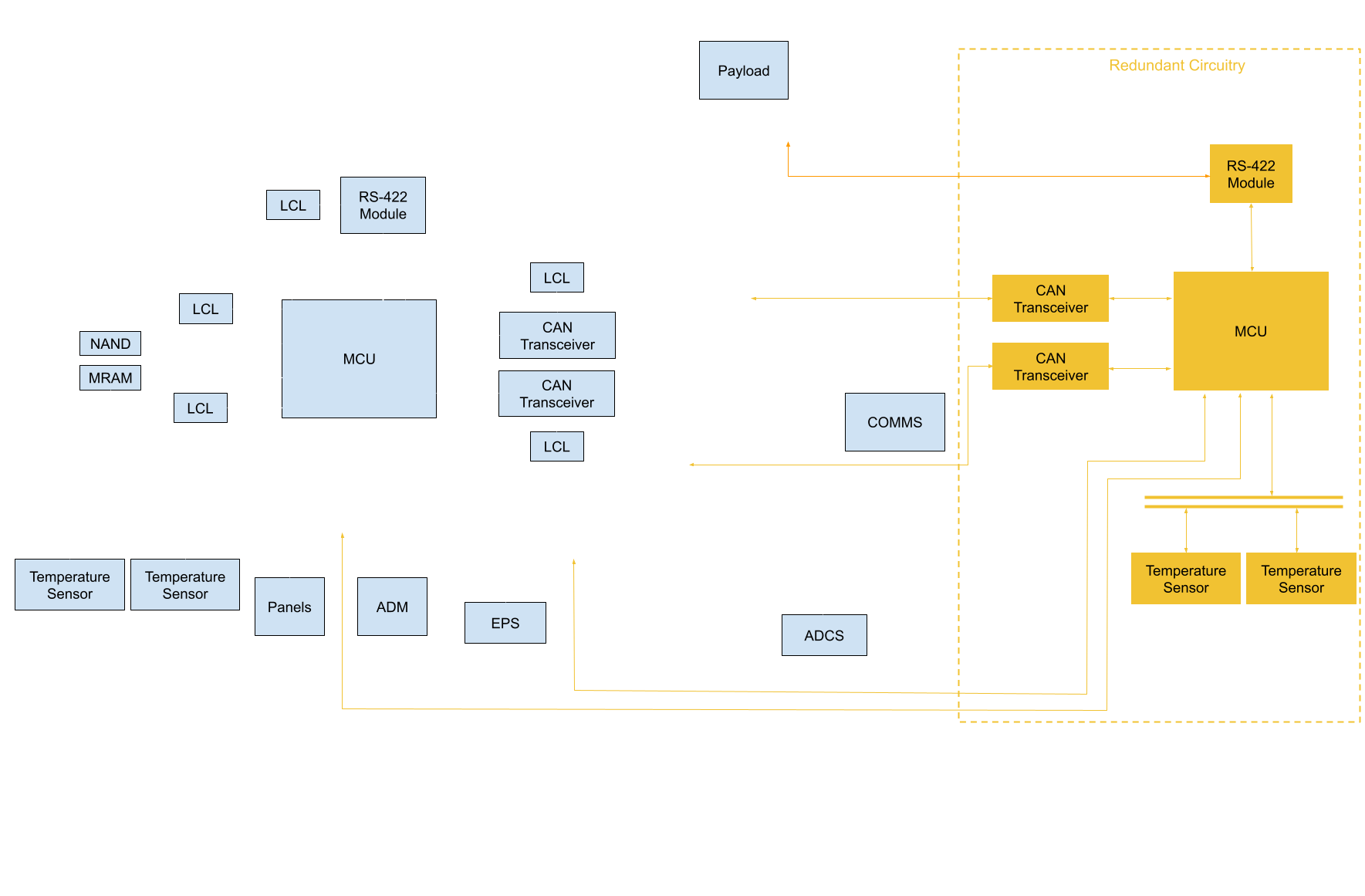
***OBC Physical Architecture***

Εικόνα που περιέχει κύκλωμα, ηλεκτρονικός μηχανικός, ηλεκτρονικές συσκευές, ηλεκτρονικό εξάρτημα

Περιγραφή που δημιουργήθηκε αυτόματα

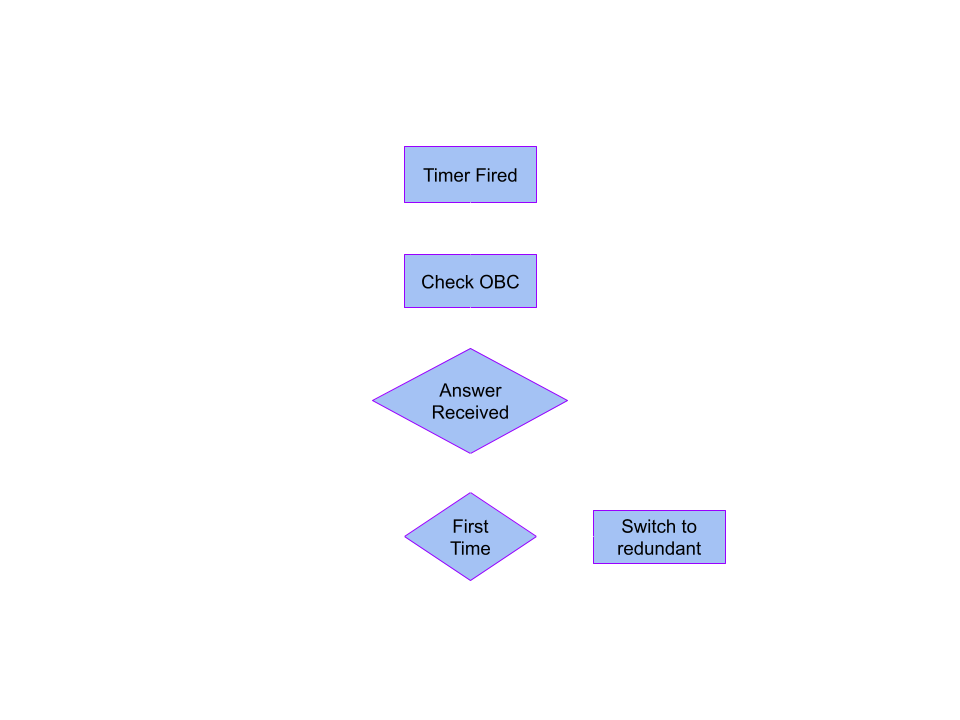
|  |  |
| --- | --- |
| * **MCU** * ATSAMV71Q21B-AAB * ARM® Cortex®-M7 * 32-bit-single-core * **PSU** * 3v3 with OC and OV protection * **Crystal** * FC-135\_32.7680KA-AG3 * 0.032768MHz ±20ppm * **NAND** * MT29F64G08AFAAAWP-ITZ:A TR * 64 Gbit (8GB) | * **MRAM** * MR4A08BUYS45 * 16 Mbit (2MB) * **CAN Transceivers** * TCAN337GD * **LCLs** * TLC555QDR * **Temperature Sensors** * MCP9808T-E\_MS |

***OBC Functional Block Diagram***



***OBC Switch to Redundant MCU logic***

* The Communications (COMMS) board MCU is interfaced with the EPS via I2C protocol. Additionally, the EPS possesses autonomous functionality enabling it to perform power cycling of the primary On-Board Computer (OBC) MCU triggered by the dedicated watchdog timer.
* Under normal operational conditions, the COMMS MCU transmits messages periodically at intervals surpassing the monitoring frequency of the EPS's watchdog overseeing the main OBC MCU.
* In the event of a malfunction detected by the COMMS, denoted by the absence of a response from the OBC for two consecutive message transmissions, the COMMS initiates corrective measures by issuing commands to the EPS. This corrective action involves the complete cessation of power supply to the primary MCU, concurrently activating the redundant MCU.



***Software Architecture – General***

The software required for the mission can be split into three parts:

* Hardware-software interface code (firmware), including:
  + The **bootloader**, covering the first instructions ran by the MCU, selecting boot partitions and resolving boot errors.
  + **Boilerplate code**, responsible for the initialization and boot sequence of the MCUs, and connecting all different software layers together.
  + **Peripheral libraries**, developed by the consortium as an abstraction and interface for external hardware components.
  + **HAL libraries** provided by the MCU manufacturer.
* System-software, playing the role of a lower application layer and resembling a "kernel", including:
  + A RTOS responsible for the **scheduling** and **coordination of parallel tasks.**
  + The **file system** used by on-board memory units.
* Application & business logic, which includes:
  + Subsystem-specific code and logic.
  + An in-house implementation of the ECSS-E-ST-70-41C PUS services, covering operational aspects of the mission.
  + Generic code and logic, including common aspects such as FDIR.
  + Miscellaneous external libraries, responsible for specific tasks.

***Software Architecture – In House Subsystems Responsibilities***

* **OBC** is responsible for:
  + **Commanding** all the subsystems.
  + Performing system-wide and COTS subsystems **FDIR** activities.
  + Switching **system modes.**
  + Handling **system data.**
  + Storing **TM data.**
* **COMMS** is responsible for:
  + **TC & TM handling.**
  + **GNSS communication** and data handling.
  + **CCSDS** data link.
* Redundancy:
  + OBC has a **redundant MCU** that takes over all OBC activities except for TM data storing.
  + COMMS will not perform TM data storing.

***ECSS Services***

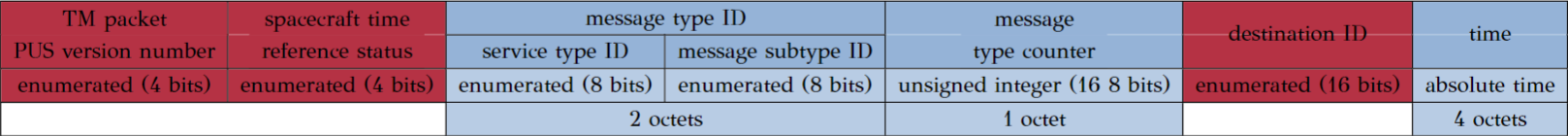
|  |  |
| --- | --- |
| * ST[01] request verification * ST[02] device access * ST[03] housekeeping * ST[04] parameter statistic service * ST[05] event reporting * ST[06] Memory Management * ST[08] Function Management * ST[09] time management | * ST[11] time-based scheduling * ST[12] on-board monitoring * ST[15] on-board storage and retrieval * ST[17] test * ST[19] event-action * ST[20] parameter management * ST[23] file management |

***ECSS Services layout of TC/TM***

* The primary headers for both TM and TC are the ones specified by *CCSDS-132.0-B-2* and *CCSDS-232.0-B-3*, with no modifications
* Space Packet Primary Header: This header is also utilized both for TC and TM and is described in *CCSDS-133.0-B-2*. It is not altered in any way in our implementation

The TM secondary header is based on the recommendation of *ECSS-E-ST-70-41C* with the deletion of some fields. In more detail:

* The **destination ID** is redundant as the same purpose is also served by the synchronization header and partly the message verification tag.
* **Spacecraft time reference** status is also left out.
* TM packet **PUS version number** has also been removed as it is a constant field.



Some fields were removed in this header as well:

* TM packet **PUS version number** has also been removed as it is constant.
* The **source ID** has been removed since it's already included in the CCSDS primary header.

The **ACK flag** is set by the operator for each TC and determines the type of acknowledgement flag to be sent as in *ECSS-E-ST-70-41C* . In more detail:

* **bit 0**: acknowledges completion whether the execution of the TC was successful or not.
* **bit 1**: acknowledges the progress of execution.
* **bit 2**: acknowledges that execution has started.
* **bit 3**: acknowledges that the corresponding packet has been accepted by the application process.

Authentication 40-bit HMAC tags are used for message authentication to ensure that the source is authorized (i.e. our ground station).

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

***High level overview of software***

Εικόνα που περιέχει κείμενο, διάγραμμα, στιγμιότυπο οθόνης, Σχέδιο

Περιγραφή που δημιουργήθηκε αυτόματα

***Operating System, Timekeeping and Commanding***

The operating system chosen by the consortium is the **FreeRTOS** operating system.

FreeRTOS in specific was chosen as a popular RTOS solution that has been extensively tested on ARM Cortex-M microcontrollers, offers an able feature set and simultaneously keeps resource utilization at low levels.

FreeRTOS supports three policies of scheduling:

* Prioritized Preemptive Scheduling.
* Prioritized Preemptive Scheduling with time slicing.
* Cooperative Scheduling.

The policy selected by the consortium is **Prioritized Preemptive Scheduling with time slicing**.

To decide the priority of each FreeRTOS Task for the business logic, the two following algorithms will be used:

* **Rate Monotonic (RM) Algorithm**: This algorithm works by assigning the priority of each task inversely proportional to its period; the shorter the period, the higher the priority. Main advantages include high efficiency in CPU utilization and ease of implementation.
* **Deadline Monotonic (DM) Algorithm**: The priority of a process is inversely proportional to the deadline. This algorithm works well with aperiodic tasks, and it can produce a schedulable system when RM fails to do so.

The above selections support future software schedulability analysis.

The **SAMV71Q21RT MCU** has a built-in timekeeping mechanism that uses its internal hardware RTC. This mechanism enables the OBC to keep track of time at any point.

To counter the loss of synchronization by drift or a system restart, the **OBC shall synchronize the time by fetching the current time from the GNSS module** on the COMMS Board.

In case of a GNSS Module Loss, the **time will be synced by TC** in every RF Pass.

For implementing high priority commands, a tailored version of the *ECSS-E-ST-70-41C* standard is used for the OBC and COMMS.

For the COTS Subsystems, an abstraction layer will be added between FreeRTOS and ECSS Services for commanding and telemetry logging, ST[02] and ST[08] are used.

***Software and Hardware***

SW-HW compatibility:

The following principles are applied:

* In-house software, except from business logic, is split in two categories:
  + In-house software modules that don’t need to depend on platform properties (architecture, endianness, peripherals, etc.): Those are implemented with portability in mind, meaning that code conforms to the language standard guarantees and those only, avoiding chip vendor interfaces. This allows for thorough testing that can run anywhere.
  + In-house software modules that must use platform specific interfaces and rely on specific properties (drivers and platform introspection code): Those are granularized and implemented in vacuum, for independent testing to be possible.
  + Software modules belonging in different categories are not allowed to depend on each other. The only exception in this rule is business logic which combines the functionality of those modules to describe the behavior of the program.
* COTS subsystems are considered to contain a black box implementation, meaning that software is always compatible with the hardware it will run on.
* Software provided by the chip vendor of in-house subsystems (language support runtime, peripheral libraries) are compatible with the hardware they run on by default and do not need further justification.
* Third party libraries, when source code is provided, are always audited by at least two developers to ensure that code is of acceptable quality, doesn’t violate any requirements and is compatible with the freestanding platform that the team wishes to use it in. In case the library passes the audit, the library must be tested to ensure that functionality meets the expectations and there are no hidden problems with the library that were not caught by the audit (for example race problems and other unprovable safety issues).

***Memory Budgets***

|  |  |
| --- | --- |
| **OBC NAND** | |
| Total Size | 8192 MiB |
| Filesystem Overhead (worst case) | x4 |
| Max bytes transmitted per pass + Filesystem overhead | 2518.56 KiB |
| Max bytes to be stored + 100% margin | 5037.12 KiB |
| Payload Data | 36 MiB |
| Min number of pass that can be stored | 1635 |
| Mean number of passes during mission lifetime | 432 |
| On-orbit generated TM that can be stored | 100 % |
| Min number of passes to be stored before erasing | 4 |
| **COMMS NAND** | |
| Total Size | 8 GiB |
| GPS Input Data | 10.08 KiB |
| Software Image | 1.32 MiB |
| Filesystem Overhead (worst case) | x4 |
| Total + 50% margin | 8.33 MiB |
| Spare | 99.903 % |
| **OBC MRAM** | |
| Total Size | 2 MiB |
| ECSS definition parameters | 8.2 KiB |
| System Parameters | 782 B |
| Software Image | 1.14 MiB |
| Total + 50% Margin | 1.72 MiB |
| Spare | 14 % |

**NAND Flash Wear:**

* Analysis shows 6.8% wear during the mission lifetime which is acceptable for the mission requirements.
* ESA Experts suggest an 8% wear of the NAND as a worst-case scenario.

***Data Budgets Conclusions***

* All R/W operations of the memories will be monitored and logged through ECSS Services (Memory Management Service). In the unlikely event of a full memory the older data will be wiped for new data to be stored.
* Based on TM size and considering that NAND is 8GB with relatively low wear, the max days the SC can go without transmitting data is **35,036 days.**
* In case of a full memory due to bad blocks and excessive wear, the policy is to delete the oldest TM to make room for the new one.
* Downlinking TMs has a time margin of 91% which is adequate.
* Downlinking TMs has a time margin (+100% margin to downlink time) of 82% which is adequate.
* Available time margin to upload software updates (compressed, +100% margin, with ECSS headers = 9,69%
* If software uplinking cannot be uploaded in 1 pass due to bad circumstances, AUTH sets a baseline of **splitting the image into 2 parts** with a deterministic way and **upload it piecewise**.

***I2C Address Table***

|  |  |  |
| --- | --- | --- |
| **Bus & Subsystem** | **Peripheral** | **Address** |
| OBC – I2C\_0 | MCP9808\_1 | 0x98 [Read] |
| OBC – I2C\_0 | MCP9808\_1 | 0x18 [Write] |
| OBC – I2C\_0 | MCP9808\_2 | 0x99 [Read] |
| OBC – I2C\_0 | MCP9808\_2 | 0x19 [Write] |
| OBC – I2C\_1 | EPS | 0x20 |
| OBC – I2C\_1 | Panels | 0x51 – 0x5F |
| OBC – I2C\_1 | ADM | 0x33 |
| OBC – I2C\_2 | EPS [Redundant Connection] | 0x20 |
| OBC – I2C\_2 | Cube ADCS | 0xAE |
| Red. OBC – I2C\_0 | MCP9808\_1 | 0x98 [Read] |
| Red. OBC – I2C\_0 | MCP9808\_1 | 0x18 [Write] |
| Red. OBC – I2C\_0 | MCP9808\_2 | 0x99 [Read] |
| Red. OBC – I2C\_0 | MCP9808\_2 | 0x19 [Write] |
| Red. OBC – I2C\_1 | EPS | 0x20 |
| Red. OBC – I2C\_1 | Panels | 0x51 – 0x5F |
| Red. OBC – I2C\_1 | ADM | 0x33 |
| **Bus & Subsystem** | **Peripheral** | **Address** |
| Red. OBC – I2C\_2 | EPS [Redundant Connection] | 0x20 |
| Red. OBC – I2C\_2 | Cube ADCS | 0xAE |
| COMMS – I2C\_1 | INA3221 | 0x40 – 0x43 |
| COMMS – I2C\_1 | TMP117 | 0x90, 0x92, 0x94, 0x96 |
| COMMS – I2C\_2 | EPS | 0x20 |
| COMMS – I2C\_2 | CubeADCS | 0xAE |
| COMMS – I2C\_4 | EPS | 0x20 |
| COMMS – I2C\_4 | Panels | 0x51 – 0x5F |
| COMMS – I2C\_4 | ADM | 0x33 |

***Binary Image Compression***

During In Orbit patching, MCU code is delivered to the spacecraft in a compressed format and therefore it must be decompressed for the software update to be successful.

Selected option for the decompression algorithm of PeakSAT is the following:

* [DEFLATE](https://en.wikipedia.org/wiki/Deflate): DEFLATE is a lossless data compression algorithm which belongs in the Lempel-Ziv (LZ) family of compression methods, optimized for decompression speed and high compression ratios. It is widely used for general-purpose compression and is part of the specification of the PNG image format. It replaces duplicated data strings with 23-bit references to the location of their first occurrence, and then encodes the resulting data into a Huffman tree, which serves as a dictionary. On the downside, its compression can be slow, compared to other algorithms. There are many implementations of DEFLATE, many of them written with extreme portability in mind.

Since PeakSat has an RF pass every 24 hours, compression speed is not critical.

***FMEA/ FDIR design – Watchdog Architecture***

* OBC monitors the subsystems health through CAN and I2C buses.
* OBC commands the EPS to power cycle any subsystem when unresponsive for long period.
* COMMS and OBC have redundant CAN bus lines while ADCS has an I2C line as an alternative.
* EPS has a redundant I2C line.
* OBC can transmit to both CAN bus line simultaneously to detect if main CAN has failed or if a subsystem needs further FDIR.
* In case OBC fails a redundant MCU is available to take over.