PixelSat I Electrical

Vinayak Vikram [Project Pixel Orbital, Stanford Online High School]

Contents

Introduction

Project Pixel Orbital is a student-led team at Stanford Online High School, aiming to launch a 3U CubeSat (PixelSat I) to sun-synchronous low-Earth orbit NET November 2026. In this paper, we document the electrical subsystems for PixelSat I.

OBC

The OBC (on-board computer/on-board controller) for the satellite is an STM32H7 running `no_std` rust. The current flight software uses the STM32′s internal independent watchdog rather than a separate external watchdog IC. Persistent non-volatile storage for items such as images and TLE data currently uses the STM32′s second internal flash bank rather than an external NOR flash chip.

For coarse sun sensing, the electrical design now uses six dedicated photodiodes, one on each body face, rather than reusing the solar panels as coarse sun sensors.

Transceiver

As mentioned in the comms overview, we are using an Ebyte E22-400T30D LoRa module as our transceiver. This module wraps the Semtech SX1262 transceiver alongside a TCXO (temperature-compensated crystal oscillator).

Pins

The module is a DIP (through-hole) package and has a 2.54mm pitch header and a SMA-K female antenna connector. The exposed header pins are:

Pin	Name	Direction
1	M0	Input (weak pull-up)

Pin	Name	Direction
2	M1	Input (weak pull-up)
3	RXD	Input
4	TXD	Output
5	AUX	Output
6	VCC	Input
7	GND	Input
8-11	—	—

The UART interface operates at 3.3 V TTL, which is directly compatible with the STM32; no level shifter is required. M0 and M1 carry weak internal pull-ups and must never be left floating. For our purposes, they are always driven by OBC GPIOs, though if using the transceiver with factory settings they can simply be grounded.

Power supply
By the datasheet:

> $2.6 - 5.5V$, 30dBm output module $\geq 5V$, 22dBm output module $\geq 3.3V$ ensures output power; exceeding $5.5V$ will permanently damage the module.

> $460 - 620mA$, instantaneous power consumption @ 30dBm

Note that the power draw of our transceiver can reach $3W$, which is about as much as all three magnetorquers torquing at once; the PDU must explicitly accommodate this as transmission frequency is quite high.

We therefore hook the transceiver up to the $5V$ rail on the satellite, separate from the $3.3V$ rail used by the STM32. This $5V$ rail must be rated for at least $1A$ to accommodate the transmit burst without sagging. To absorb the instantaneous current draw during transmission without causing a voltage droop that could affect other subsystems, we must place a capacitor across the VCC and GND pins (close to the module headers, as the impedance in the traces will hinder the effectiveness of the capacitor). Probably best to measure the TX power draw and decide on the capacitor empirically, after basic circuitry is worked out. For now, a $200\mu F$ capacitor should suffice. Notably, by the manual hardware design specifications, no signal or power traces should be routed beneath the module footprint.

The RX current is approximately $13mA$, and the deep sleep current (probably never used) is approximately $3\mu A$, which are both negligible relative to the TX current.

Mode control
The module's operating mode is determined by the logic levels on M0 and M1. The four modes are:

Mode	M1	M0	Description
0	0	0	Normal transmission
1	0	1	WOR (wake-on-radio)
2	1	0	Configuration

Mode	M1	M0	Description
3	1	1	Deep sleep

After a mode switch, the module takes approximately 9-11 ms to transition; the AUX pin indicates busy (low) during this period. The STM32 must wait for AUX to return high before issuing any further commands or data.

AUX pin

AUX is an output from the module indicating its internal state: high means idle, low means busy. STM32 reads AUX as a GPIO input; a $10k\Omega$ pullup should be placed on the STM32 side and the pin connected must be configured at boot to be a high-impedance input pin. This is because if AUX is driven or pulled low externally for more than one second during the boot window (approximately the first 5-16 ms after VCC is applied), the module forcibly enters firmware upgrade mode and will not operate normally. The AUX signal is used for two purposes. First, as a handshake before transmitting; the STM32 must confirm AUX is high before writing data to RXD, as writing while the module is busy risks data loss. Second, as a receive indicator; when an over-the-air packet is received, AUX briefly pulses low and then returns high once the packet has been output on TXD.

Reset

The E22-400T30D does not expose a hardware reset pin; this exists only on the T30S variant. As such, we will have two software-level reset mechanisms. The first is the `AT+RESET` AT command, issued in configuration mode (M1=1, M0=0); the module acknowledges with `+OK` and resets approximately 30 ms later. The second is a hardware power cycle: a MOSFET on the $5V$ rail allows the STM32 to cut and restore power to the module entirely. It might even be prudent to only initialize the module after the STM32 boot (pull down the MOSFET to GND).

Antenna

The antenna connector on the module is a 50Ω female SMA port. For testing purposes, we connect it to the ANT-433-OC-LG-SMA antenna, a commercial omnidirectional half-wave dipole whip antenna designed for 430-435 MHz LPWA (low-power, wide-area) applications (such as LoRaWAN). Note that we will be using a tape-measure antenna on the actual deployment.

Magnetorquers

math stuff

We first derive the equation for the moment given by one of our torquers. Assume we are using magnetic wire with resistance per unit length λ, bus voltage V, number of turns N, and cross-sectional area $A = s^2$. The equation for the mangetic moment given by a single magnetorquer is given by $m = N \cdot I \cdot A$. Substituting $I_0 = \frac{V}{R} = \frac{V}{\lambda l} = \frac{V}{4N\lambda s}$ into this, we see that the total moment of the torquer is $m = N \cdot \frac{V}{4N\lambda s} \cdot s^2 = \frac{Vs}{4\lambda}$ (assuming squared corners; we will probably have more like $\frac{Vs}{3.8\lambda}$ in reality but that is insignificant at the moment).

Given the assumed $3W$ combined power ceiling, it is prudent to throttle each magnetorquer at $1W$. As such, by $P = \frac{V^2}{R}$, we get by $R = \lambda 4Ns$ that $N = \frac{V^2}{4s}$. For now, take our bus voltage to be $5V$ (subject to change).

design

gee gee gee see fusion, however by equations we have estimated that torquers should be around idk 8cm/side. can be a bit less probably, since we have rounded edges, but only marginally.

wire diameter 0.066mm, very doable. shaping up to have $s \approx 0.08m$, at least 80 turns. probably have 100 turns or something

As of right now, each torquer will be constructed by winding copper wire around a central rectangular bobbin with guardrails to make sure the wire does not slip off the top/bottom.

control

simple open-loop PWM would not be acceptable in this context since resistance increases as the wire heats up as well as the fact that the battery fluctuates, so we would need to implement closed-loop PWM with a current-sensing resistor or a digital monitor of some sort. Obviously, the STM32 cannot send 11.1V over a bus, so we use an H-bridge (a combination of MOSFETs) to allow the STM32 to flip the 11.1V from the battery across the coil to generate our torque.

Sources

https://www.cdebyte.com/pdf-down.aspx?id=4217 (Ebyte E22-series manual)
https://www.te.com/en/product-CAT-ANT0028.html (Linx ANT-433-OC-LG-SMA datasheet)