The Pirate Beeper

R&D Documentation

R&D Project – Confidential document
Classification: PIRATE SECRET

ph0wn CTF 2026

R&D Request: Pirate Beeper

Context

Captain **Pico**, fearsome pirate of the digital seas, has submitted an R&D request to his crew for the design of a **tactical communication beeper**.

The objective: allow Pico to transmit orders remotely to his first mate, **Gerboise**, without using conventional radio channels that are easily intercepted.

Beeper Features

The beeper has several command functions:

Command	Description
`pico_attack`	Attack order. Gerboise launches the assault on the target ship. The screen displays the naval battle scene.
`pico_boom`	Sabotage. Triggers the detonation of charges placed on the enemy vessel. The screen confirms the explosion.
`pico_home`	Reset the beeper to its initial state.

Each command is transmitted over radio and decoded by the beeper, which displays the corresponding screen on the embedded display.

Secret Feature

REDACTED

Technical Specifications

Hardware Platform

The beeper is built around the **STM32H573I-DK** board from ST Microelectronics, featuring an ARM Cortex-M33 microcontroller with TrustZone.

A **CC1101** radio module from Texas Instruments is connected via SPI to handle sub-GHz communications.

Component	Details
MCU	STM32H573IIK3Q – Cortex-M33 @ 250 MHz, 2 MB Flash, 640 KB RAM
Display	LCD 240x240 RGB565
Radio	CC1101 – Sub-GHz transceiver (300–928 MHz), configured at **433.92 MHz**
Interface	SPI + GPIO (GDO0 on PG15)
OS	Zephyr RTOS v4.3

Radio Characteristics

The beeper uses **OOK** (On-Off Keying) modulation on the **433.92 MHz** frequency.

Commands are encoded using **PWM** (Pulse Width Modulation) with a fixed bit period of **1212 µs** and the following parameters:

Pulse type	Duration	Meaning
Short pulse	376 µs	Bit 1
Long pulse	780 µs	Bit 0
Sync pulse	2209 µs	Message delimiter

The CC1101 is configured in **asynchronous serial mode** (`IOCFG0 = 0x0D`), which allows receiving the raw OOK signal on the GDO0 pin. The firmware measures the width of each pulse through GPIO interrupts to decode the bits.

Message Format

Commands are transmitted in **ASCII**. Each byte is sent **MSB first** (Most Significant Bit first), meaning bit 7 is transmitted first and bit 0 last. Each message is repeated **2 times** consecutively to ensure reception reliability. The firmware only validates a command if both copies are identical.

⚓ **End of document – Yo Ho Ho!** ⚓