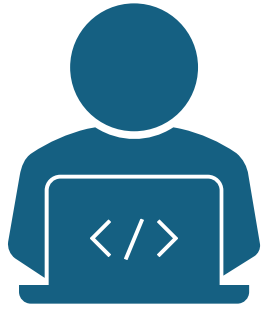


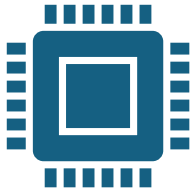
“Microcontroller Projects and Prototyping with QPC Framework”

Kamurasi jordan Arthur

purpose of the presentation:

- Showcase my interaction with the provided materials (STM32f407, STM32 NUCLEO-C031C6, PIC16F15244 Curiosity Nano, TM4C123GH6PM).
- Proof of capability of use of C language effectively for microcontroller projects.
- Demonstrate familiarity with the real-time event-driven framework (QPC).

Tools used



Microcontroller Boards:

STM32f407

STM32 NUCLEO-C031C6

PIC16F15244 Curiosity Nano

TM4C123GH6PM

Development Environments (IDEs):

TI Code Composer Studio

VS Code with PlatformIO extension

STM32CubeIDE

MPLAB X IDE

Tools for State Machine Design:

QM tool (auto-code generation for QPC framework)

The PELICAN State Machine

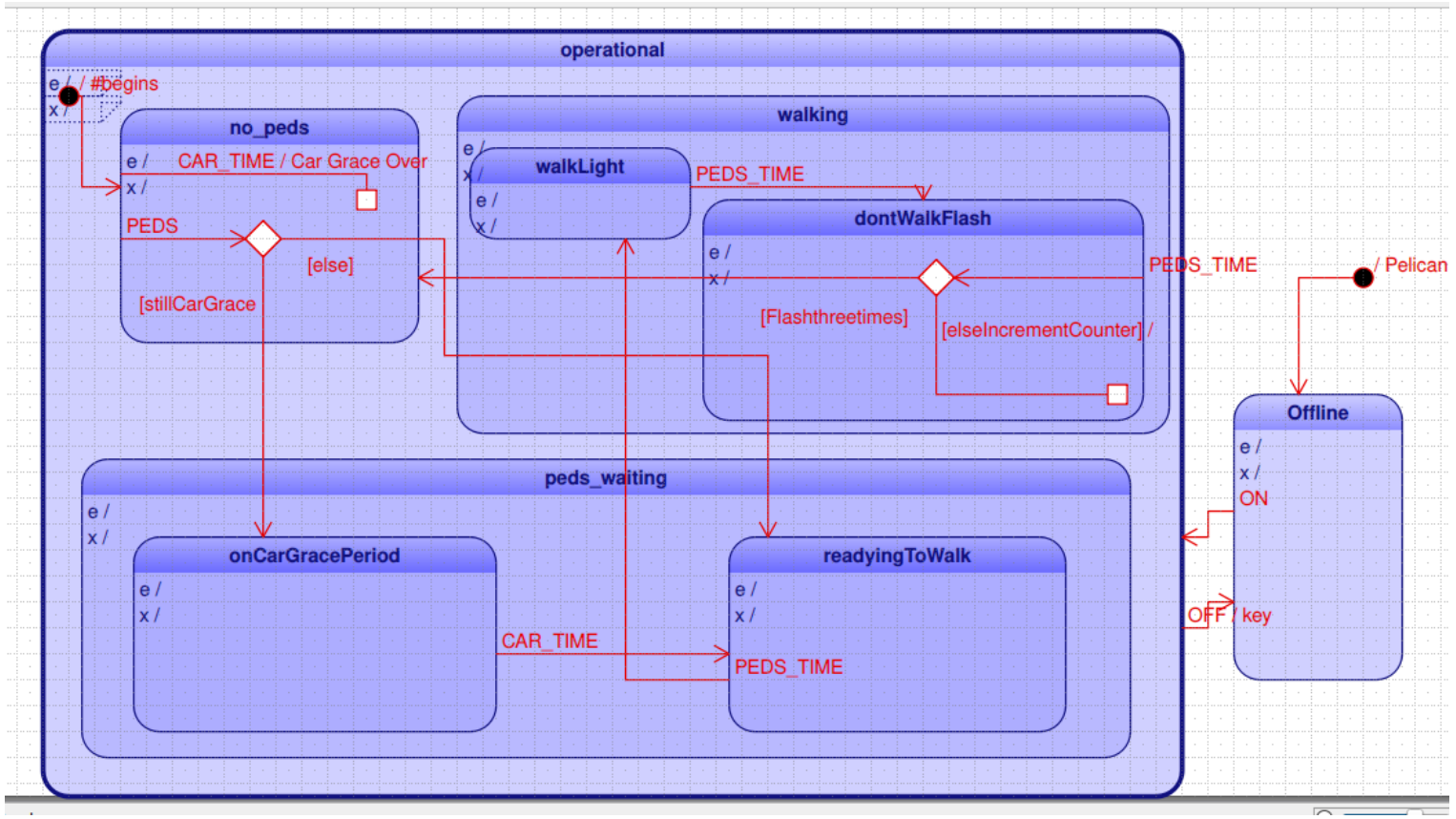
Posix based example.

Brief

Prioritizing pedestrian and vehicle safety, the system begins with cars enabled (green light) and pedestrians disabled (“Don’t Walk” signal). When a pedestrian presses the crossing button, cars receive a yellow light, then red, and pedestrians get the “Walk” signal. This changes to a flashing “Don’t Walk” before reverting to green for cars. After each cycle, cars maintain a minimum green-light period. An operator can take the system offline anytime, displaying flashing red for cars and flashing “Don’t Walk” for pedestrians, and can restore online operation at will

Illustration

{state machine diagram}



Smart Irrigation System

A mimic of an optimized water usage control system in agricultural fields using the *TM4C123GH6PM* micro-controller.

Description.

- Imagine a smart irrigation system designed to efficiently manage water usage in an agricultural field. The system starts in an idle state, constantly monitoring soil moisture levels and weather forecasts. When the soil moisture level falls below a predefined threshold, the system transitions to a watering state, activating irrigation valves to water the crops. If the weather forecast predicts rain, the system delays irrigation to conserve water. Users can manually control the irrigation system, overriding automated processes, or set a schedule to water the crops at specific times. In the scheduled watering state, the system begins watering at the scheduled time and adjusts based on weather forecasts. An operator can put the system into maintenance mode for servicing, disabling all automated and manual watering events. The system ensures efficient water usage, prioritizes crop health, and adapts to environmental conditions, offering seamless servicing options without disrupting normal operations.

Project was un-fortunately deleted.

Elevator Operation System

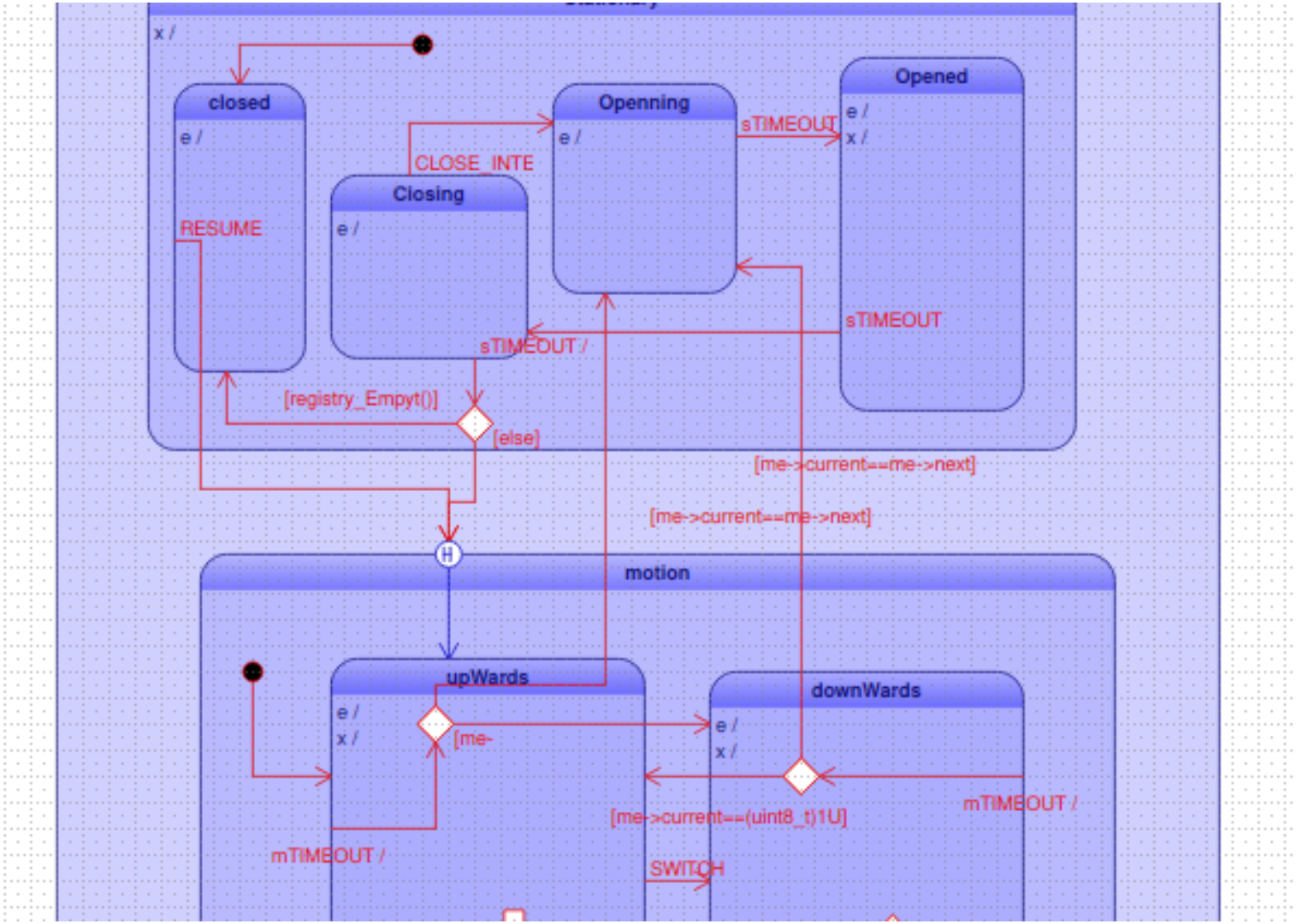
System designed for efficient elevator movement using anSTM32F407 Discovery board

Implementation Details

- **Tools Used:**
  - STM32F407 Microcontroller.
  - VS Code with PlatformIO extension.
  - QPC Framework for state machine implementation.
- **Design description:**
  - Prioritizing user requests, efficiently serving multiple stops in one direction, opening doors at all entry and exit points, and switching direction once no further requests remain or the elevator reaches the top. The system also hibernates to conserve power when no requests are active.

Illustration

{state-machine diagram}.



Audio Player System

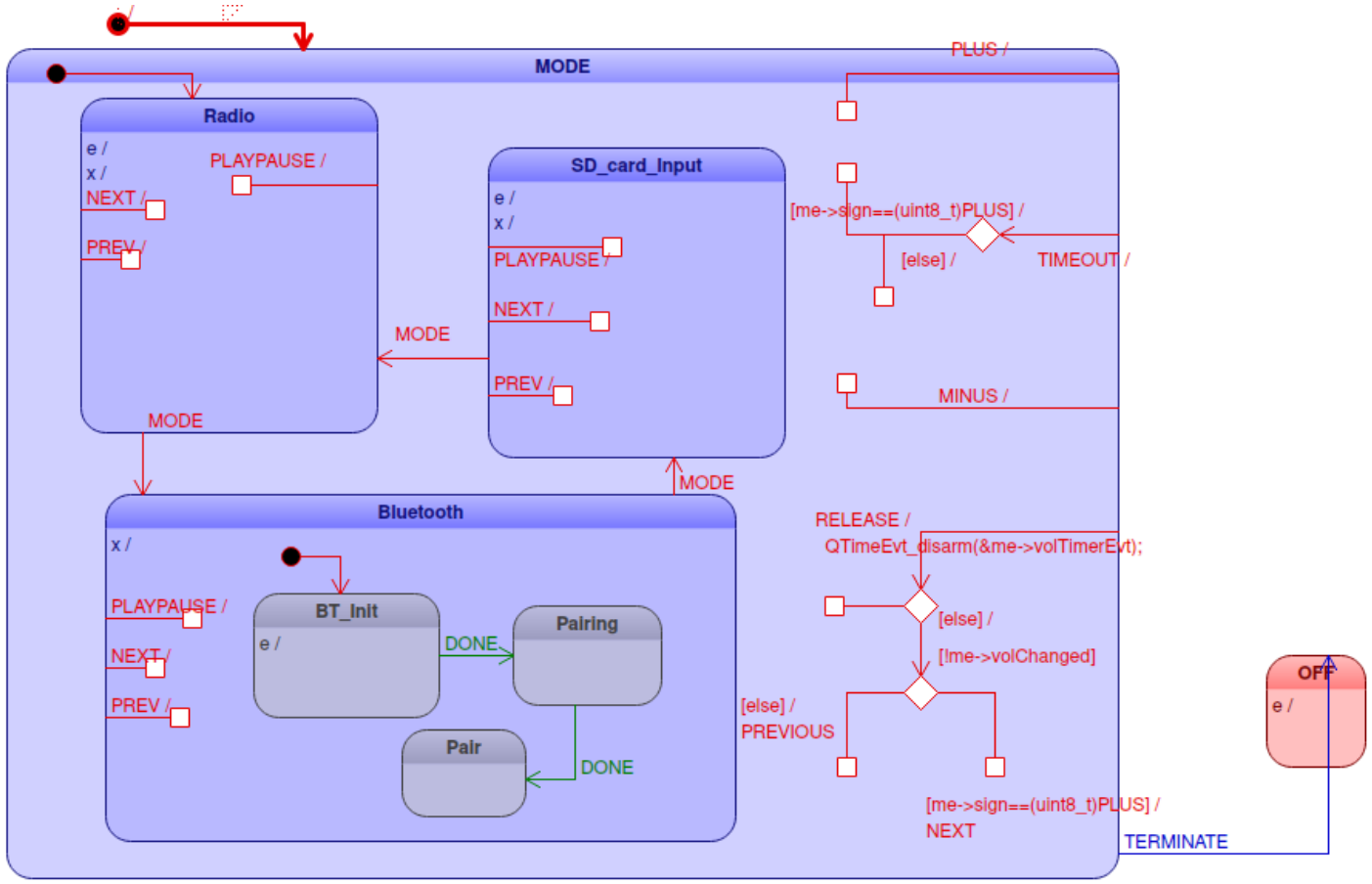
Control system operations of a generic audio player with integrated radio and MP3 functionality designed.

Implementation details

- **Tools Used:**
  - STM32 NUCLEO-C031C6 microcontroller.
  - STM32CubeIDE for design and implementation.
- **Design description.**
  - The audio player system manages controls for essential operations, including play, pause, and volume adjustment. It integrates radio and MP3 player functionalities to provide a seamless multimedia experience. Designed for intuitive and efficient user interaction.

Illustration

{state-machine diagram}



Glowing Bulb - PIC Microcontroller

- **Objective:**
  - A simple project created to explore and familiarize with the PIC microcontroller and **MPLAB IDE**.
- **Brief Overview:**
  - Used *Purse Width Modulation* **PWM** to continuously glow the user-led with **different duty cycle** settings in-order to bring out the impression of a glowing bulb.
  - Utilized MPLAB's Code Configurator **(MCC)** tool to facilitate development and configuration.
  - The **PIC16F15244 Curiosity Nano** does not have a QPC port currently.

Issues points.

Q-SPY SOURCE FILES ISSUES REGARDING THE MACHINE

THE FREE VERSION OF QPC - QXK DOES NOT CONTAIN QXK SOURCE FILES

REGISTERED MISS BEHAVIOR FOLLOWING COMPUTER REPAIR

END