

Figure 1: Overall system design

## 1 The Control System

This section describes the embedded control system and user interaction. The latest system software and documentation can always be found on github at <https://github.com/hedj/fusion/>.

Flowcharts in this section use the visual language from the DRAKON project. See <http://drakon-editor.sourceforge.net/language.html> for details.

### 1.1 The Control Computer

The control computer is a standard PC presently running windows 10. It shares its connection to the internet via ethernet, which is connected to a hub. Oscilloscopes and the IRC server hang off this hub. At present, all the software agents described later run on the control computer, although in principle it would be possible to run many of them on other computers.

### 1.2 The Bank Controller

The bank controller is a Controllino Mega; an Arduino-based programmable-logic controller. The bank controller is in charge of the following functions:

- Capacitor-bank and HV pulse generation

- HV voltage control
- Capacitor bank charge control
- Audible alert before pulse

The top-level flow of the bank controller is shown in Fig. 2.

The serial protocol used is nonstandard; a definition can be found at <https://github.com/hedj/ns1c>. It is designed to facilitate debugging using a serial terminal; ns1c is a simple frame-based protocol using line-feeds as end-of-frame markers and whitespace for escape. It is also checksummed for reliability. The particular checksum is only 8 bits in size, and is optimised for speed of implementation.

The most important thing to emphasise is that the controller is implemented as a tight loop reacting to the change in time. There are no blocking wait statements; in this way we ensure that input can continue to be received from the front panel switches or serial as the process runs. This allows the safe abort of the system regardless of its state, which is rather important.

The two operations which consume a reasonable amount of time and are thus implemented as conditional actions in the inner loop are pulsing and charging, respectively Figs. 3 and 4.

### 1.3 The Stepper Controller

The stepper controller is an Arduino UNO; four pins and mosfets are used to switch an externally-supplied 12V to the poles of a stepper motor. The stepper motor accepts simple serial commands; e.g 'F5' for forward 5 milimeters, or 'R10' for 'Reverse 10 mm'.

### 1.4 The IRC Server

The IRC server is a raspberry pi running Linux. At present it has the fixed IP address 192.168.137.3. The IRC server runs NGIRCD, an open-source IRC daemon. All user-facing software connects to this IRC server. In this fashion, we can easily view and log all messages and commands being issued or responded to by the various subsystems.

### 1.5 The Agents

Each important subsystem has its own IRC bot; there is a bankbot which translates commands and messages for the bank controller, a stepperbot for the steppers, and so on. There are also some utility IRC bots. The talkbot does text-to-speech translation, and the sequencebot can perform complex sequences of tasks, including sending messages to the other bots.

The complete list of bots (Agents) presently in use is as follows:

- bankbot : controls the cap bank and HV power supplies

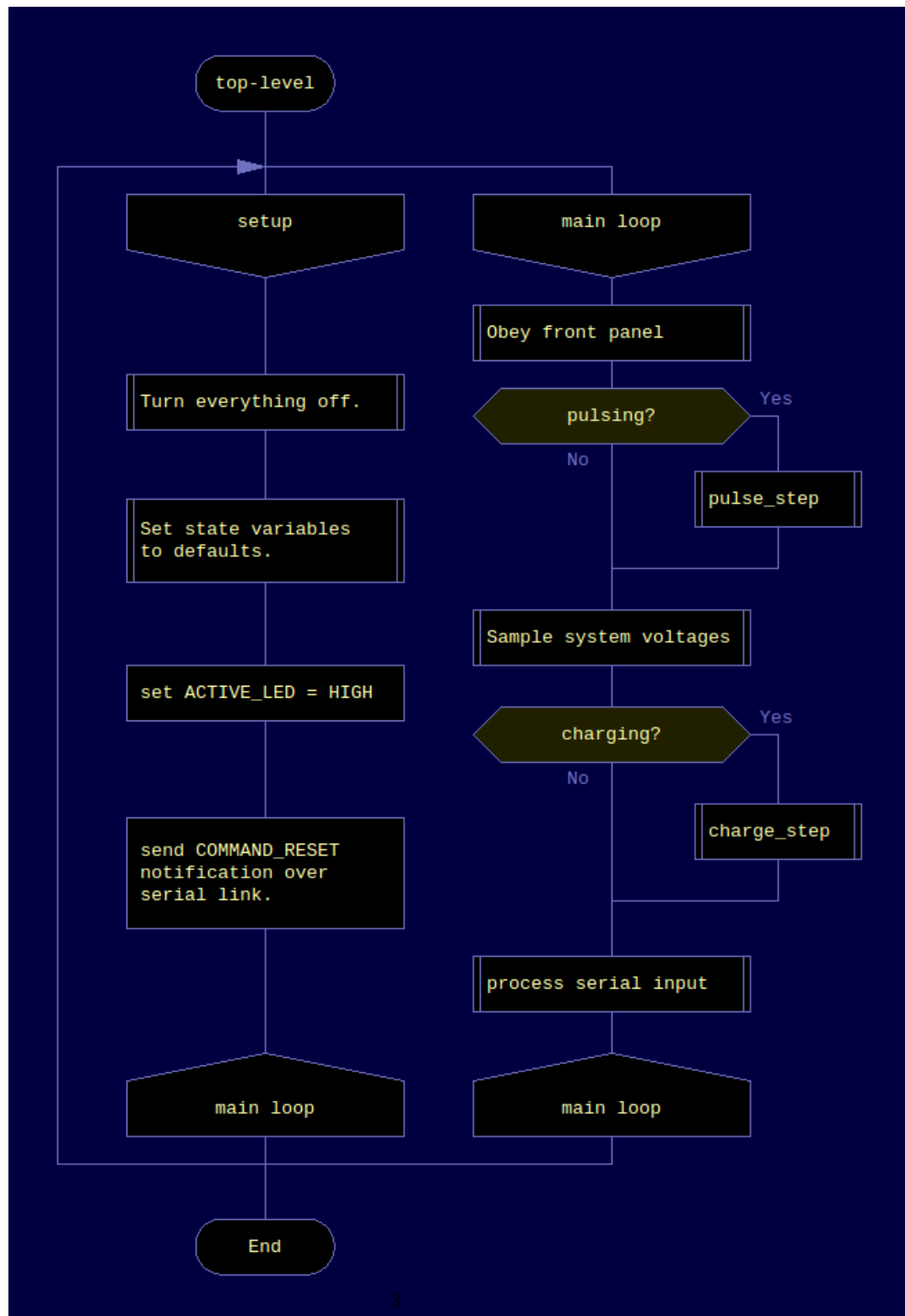


Figure 2: Top-level flow of the bank controller

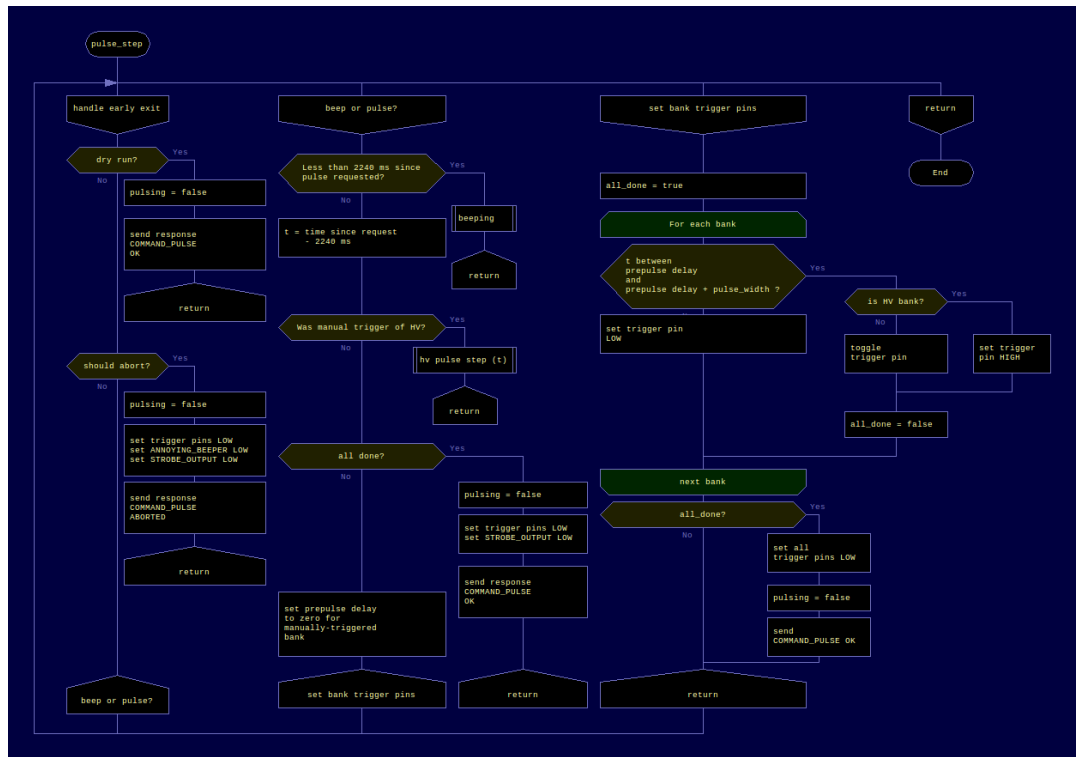


Figure 3: Pulse Step processing

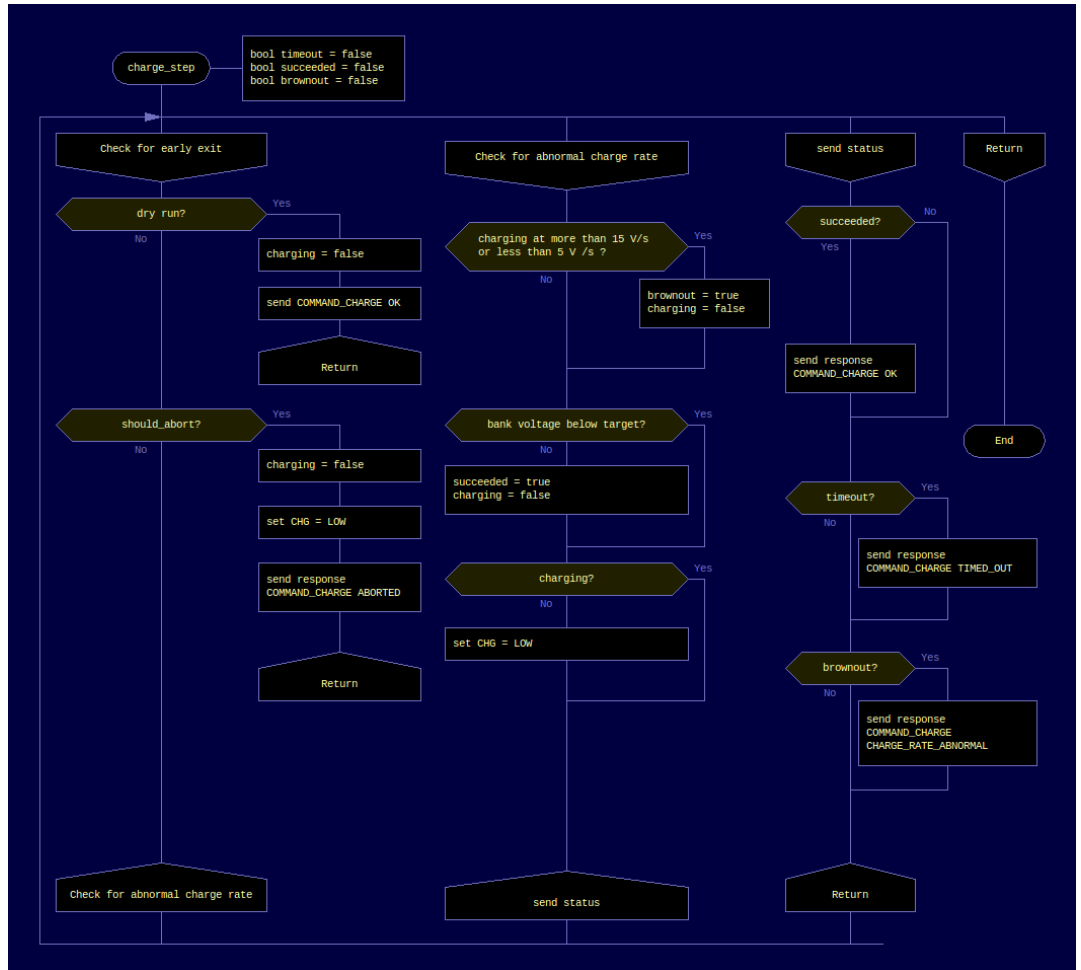


Figure 4: Charging step logic

- ipfsbot : uploads files or directories to IPFS
- picobot : controls the Picoscope 3405D oscilloscope
- rigolbot : controls the two DS10x4z oscilloscopes
- talkbot : says things out loud - e.g countdowns
- robot : runs complex sequences of commands

## 1.6 Using the System

The HEXCHAT application is set up to connect to the IRC server and become the system operator; it then connects to the channel called `#system`. The `run_bots.bat` batch-file (linked to on the desktop) will invoke all of the bots, which will then initialise themselves and connect to the same channel.

Sending any of the bots a message of the form “BOTNAME: help”, e.g ‘bank: help’, will cause that bot to reply with a description of all the commands supported by that bot. To issue a command CMD, you can just type “BOTNAME: CMD”.