PowerPmac IDE Investigation

Intro

This investigation was an attempt to verify if it is viable to put a Power PMAC IDE Project under version control and cope with changes to the brick made by 3rd party tools using the sshgpascii interface. That is to say is it possible to manage the configuration of these devices using a similar approach to that which Diamond has for the Turbo PMAC2.

The short answer is No. It is not possible to use 3rd party tools with this device in combination with the IDE.

However I now understand the life cycle of configuration items enough to recommend some changes to the Omron Software that would make it possible.

These notes are from IDE v4.4.1.7

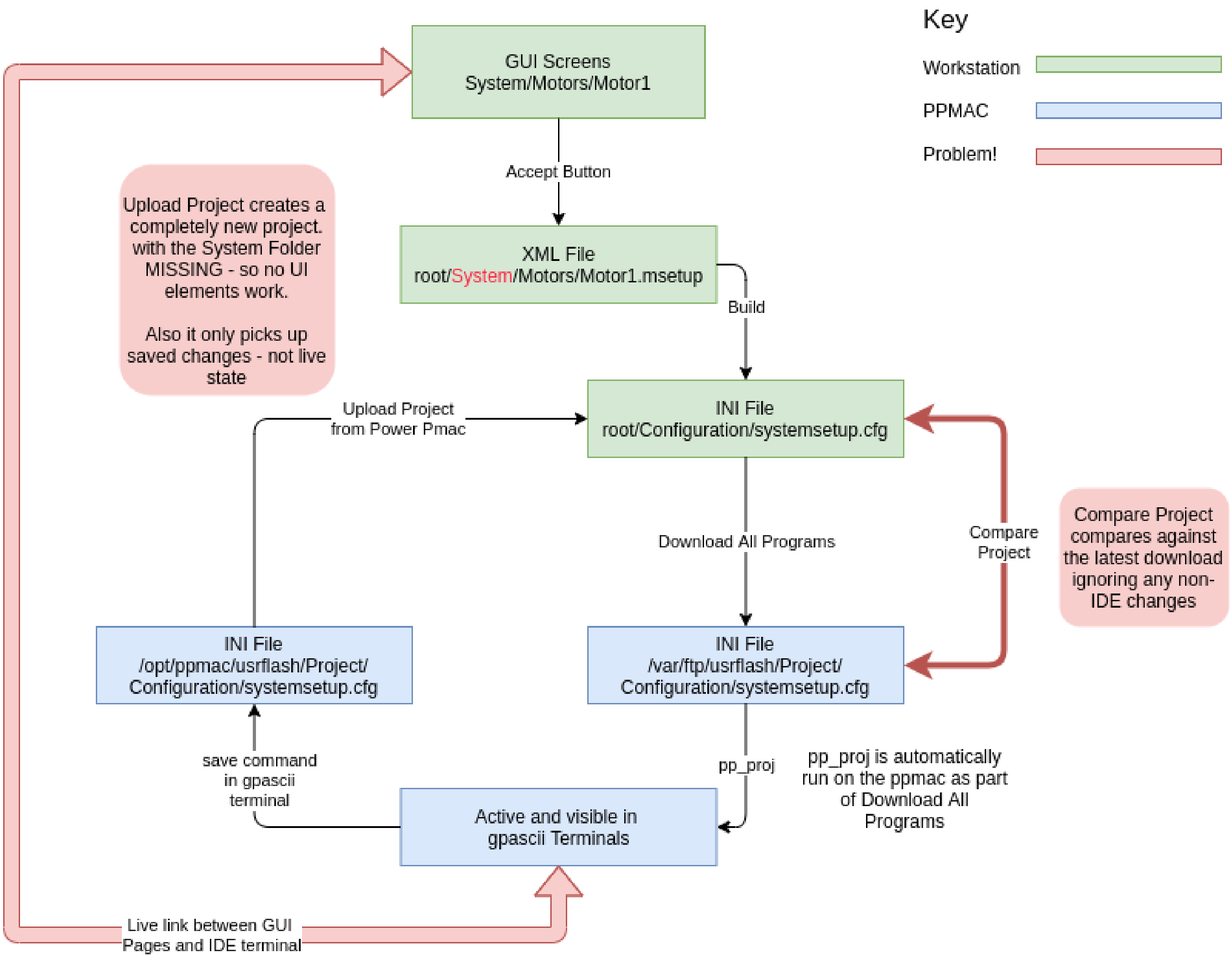
Life cycle of a configuration item

For this investigation I took a very simple value Motor[1].JogSpeed (I122) and followed what happened when it was changed from various places. It can be changed:

1. From the GUI screen accessed via System/Motors/Motor1 Commissioning Velocity Screen
2. Via modifying the value in the IDE's connected Terminal
3. Via modifying the value in a standalone terminal (or 3rd Party Tool)

The value's life cycle is summarised in the following diagram.

Power Pmac IDE: Lifecycle of Motor[1].JogSpeed AKA I122



Upload Project creates a completely new project. with the System Folder MISSING - so no UI elements work.

Also it only picks up saved changes - not live state

Compare Project compares against the latest download ignoring any non-IDE changes

pp_proj is automatically run on the ppmac as part of Download All Programs

Changes in the GUI pages immediately alter values in the live state (when Accept is clicked). It also writes the changes to the relevant System/*/*setup file. This is not reflected in the remaining project files until Build and Download is performed.

Changes via terminal are reflected in the GUI because each GUI page reads its values from live state as it is opened - but note that this means what is displayed is out of date with respect to the System/*/*setup file, and worse - you cannot update that file by doing a save because the IDE does not know the page is dirty. Hence any changes via terminal are lost next time a Build and Download is performed.

When you use the Add Motor context menu a new MotorX.msetup file is created. I assume that at this point all values are read from the current live state. This implies it is possible to re-create the GUI XML files from current state but that at present there is only a manual way of doing so.

Issues

The issues with respect to version control and 3rd party tools are below. The term 3rd party change here means that a connection to gpascii was made and commands were executed that modified state.

1. The overriding issue is that there is no way to get from a 3rd party change to updated GUI XML files. This means we can't have 3rd party tools and use the GUIs. This even affects engineers who tweak values in the IDE's own terminal - although the GUI sees these changes the Project files do not get updated to reflect them.
2. The comparison feature in the IDE only compares with the most recent download and therefore never considers any 3rd party changes.
3. The System folder is omitted from Project Download. Thus when I Download and then Upload a project I'm missing all of the GUI XML files and there is no easy way to reconstruct these files.

4. The project download does NOT omit hidden files in the root of the project folder and hence ends up sending the entire git repository to the PPMAC and then includes it in future comparisons
5. The binary solution file gets updated at unexpected times and dirties the repository (not a major issue)
6. The pp_proj is sensitive to the existence of at least one file in the Logs folder and fails completely if it does not exist. It is therefore not possible to have the repo store only source files, to get around this I have stored all the files generated by build and download. This is not ideal because again this dirties the repo history. The full set of essential files is not know.

Other Issues:

1. The IDE is very sluggish even on a AMD® Ryzen 7 5800x 8-core processor × 16 with Nvidia RTX 3090 and 32GB RAM. UPDATE: I am using a ppc based clipper and have been informed that the ARM devices are faster.
2. It will crash repeatably with the following sequence
    a. Create blank project
    b. download to brick
    c. save and exit
    d. open project by double clicking the solution file.

Good Stuff

The following improvements are noted since the last investigation around 18 months ago.

1. Download to brick is reliable and did not hang when running locally and connecting to Power Clipper over VPN, this used to result in a hang
2. Upload is also smoother and does not require you to create a project first
3. The comparison feature makes it much easier to decide what to do when there is a mismatch between project and ppmac

Proposed Fixes

A few changes to the IDE should overcome all of the issues discussed here:

- Provide a mechanism to refresh all of the System/*/*setup files against the current live state of the brick
    - I perceive that this is how a new Motor?.msetup file is generated for instance so it should be easy to invoke this code for each of the existing *setup files
- Change the compare function to work against /opt/ppmac/usrflash instead of /var/ftp/usrflash
    - Or even better make it possible analyze the situation when files are no longer in sync by optionally providing comparison with:
        - The last download in /var/ftp/usrflash
        - The last saved state in /opt/ppmac/usrflash
        - The current live state by doing a save to a temporary area and comparing with that.
- Omit hidden files from project Download
- Document a clear distinction between source files and output files and ensure that pp_proj is not dependent on download of transient files.