Propeller programming Assembler, Spin, and Test Driven Development

Sridhar Anandakrishnan

November 10, 2016

Contents

List of Figures

Listings

Preamble

This book is intended for those who are famililar with Spin programming for the Parallax Propeller but have an interest in learning Propeller C and Propeller Assembler (PASM) programming. The overall task we will pursue in the book is to implement a Delta Compression algorithm: first in Spin, then in PASM, then in C. Along the way, I talk about Test Driven Development and will end with a chapter on hardware manipulations.

Intended audience

It will be helpful to have some knowledge of the Spin programming language. The intent is to help you extend the capabilities of the Propeller processor by using the Assembler language. If you don't know Spin, but do know another programming language (C or Python, for example), you will still be able to follow along effectively.

You can only learn by doing, so you must purchase a Propeller board such as the Quickstart board[1] so that you can run the code.

Formatting

Code listings are typeset in a typewriter font: `nSamps := 1`.

In order to keep the bloat down, I often elide lines that have been explained earlier. I will insert an ellipsis to indicate that: (`...`).

Lines of code in Spin may not have a line break. When a long line listing is broken into two because of the page width, this is indicated with a red arrow: (↪).

The code examples can be downloaded from github (`https://github.com/sanandak/propbook-code`). There are two libraries (`FullDuplexSerial4PortPlus_0v3` and `Numbers`) used in the code. Both can be downloaded from the Parallax Object Exchange (`obex.parallax.com`) and for convenience I include them in the github repository.

The book is typeset using the LaTeX text processing macros. I encourage you to take a look at the `tikz` package for drawing, the `bytefield` package for showing bits and bytes, and the `listings` package for formatting code. The start of my latex file looks like this:

```
1 \documentclass[usenames,dvipsnames]{scrbook}
2
3 \usepackage{bytefield}
4 \usepackage{tikz}
5 \usetikzlibrary{matrix,positioning,trees}
```

[1] `https://www.parallax.com/product/40000`

```
\usetikzlibrary{shapes.multipart}
\usetikzlibrary{shapes.geometric}
\usetikzlibrary{chains}
\usepackage{listings}
```

Trains

As you will discover, I like trains! But my choice of trains as an analogue for the Propeller isn't entirely arbitrary: like the Propeller, trains stations have many parallel tracks with trains moving at different speeds and performing different tasks... but with the need to somehow communicate between each other and with the outside world.

And there is always the possibility of a crash!

Gentle Programmer, as in any train station, you will experience tears and heartache enroute to your destination, but if you persist, there is a great reward at the other end![2]

[2] Oh, if only Rick and Ilsa had used github! (What? You don't know who they are? Put down this book and go rent *Casablanca*!)

Cover photo: Double slip at Munich Central, 2005. Photo by Björn Laczay, license CC-BY-SA 2.0. `https://commons.wikimedia.org/wiki/File:Double_slip_at_Munich_central.jpg`.