CPUの理解を容易にするシステムと解説サイトの構築
―ファミコンを解剖してバイナリに立ち向かえ。あと時を司るスクリプト言語。―

1. 背景

2013年の現在では、ハードウェアを抽象化したRubyやJavaScriptによるWebアプリや、Javaによるクロスプラットフォーム開発が活発であり、「学習の高速道路」とも例えられるほど、無数の情報が流通し、プログラミングの敷居が下がってきている。

しかし、逆にアセンブラやメモリ、IO、バスなどの、機械としてのコンピュータの情報を見かける機会は減少し、プログラマからすらもコンピュータはブラックボックスとなってしまった。

さらに近年では電子書籍が注目されている。コンピュータ上で動くことを生かしインタラクティブ性を売りにした電子書籍なども存在するが、それらはタッチすると動画が再生される程度のものが多く、コンピュータ上で動く事を生かしきれていない。

2. 目的

本プロジェクトでは任天堂のファミリーコンピュータ、通称ファミコンを用いて、アセンブラやメモリ、IOなどを理解できる電子コンテンツと、そのためのインフラを作成する。

3. 開発の内容

3. 1 バイナリから目覚めるぼくらのファミリーコンピュータ！[1]

今回開発したエンジンを用いて図1のようなソフトウェアを作成し、1月中旬には兵庫県の高等学校でワークショップを行った。

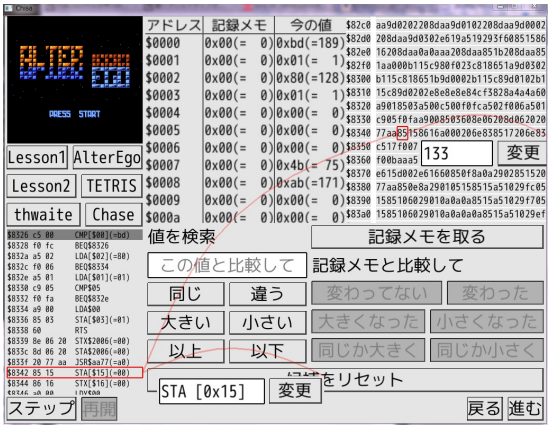

図1: メイン画面

このコンテンツではファミコンの命令実行、メモリの一覧の参照、メモリの検索、そして任意のメモリの書き換えなどができる。

今回のワークショップでは、このコンテンツを用いて「コンピュータは計算機である」こと、「キャラクタや座標などもゲーム内から直接は見えないが数値で管理されている」こと、そして当然「メモリ上の数字を書き換えれば、ゲームのルールを無視したような動作をさせることも可能である」こと、などを学び、コンピュータはブラックボックスでも何でもない、という事を伝えた。

---

1 http://ledyba.org/famicom/ バイナリから目覚めるぼくらのファミリーコンピュータ！

3. 2　ど～なっつ[2]、時を司るスクリプト言語

紙の本の利点の一つに、読み進めていくうちに分からなくなってしまったら、一旦すこし戻ってもう一度読み直し、理解を深めることができる点がある。

一方、コンピュータ上でコンテンツを作ることの最大のメリットは、プログラムを記述することで、静的な本とは違ってユーザーの動きに対応した動的なコンテンツを作れる事にある。現在の大半の電子コンテンツは XML や JSON などの静的なデータで制御フローまで含めて作られているが、今回はコンテンツの制御をプログラムとして動的に記述した。

しかし、そうしてしまうと通常コンピュータ上で動くプログラムは逐次実行であるため、今度は元に戻す事はできなくなってしまう。そこで今回、**言語処理系外の環境まで含めて実行状態を時間操作**できる言語、「ど～なっつ」を開発した。

3. 3　時を司る GUI ツールキット、ちさ[3]

上記のど～なっつを用いてコンテンツを作るための GUI ツールキット「ちさ」を OpenGL を用いて開発した。プラットフォームに依存しない設計であるため、Windows、Linux、Android でのクロスプラットフォームを実現している。

ファミコンの動作をリアルタイムで伝えるため、通常 10～20FPS で動作する GUI ツールキットよりも高速な描画を行う事ができ、PC 上では 60FPS 以上を、Android タブレット上でも 50FPS を達成している。

バックエンドに「ど～なっつ」を用いているため、今回のワークショップで用いたものなど、「ちさ」上で作成したアプリケーションは全て、過去に戻ったり、そこから再度時間移動したりするような「時間操作」を行うことができるようになる。

4.従来の技術（または機能）との相違

4. 1　バイナリから目覚めるぼくらのファミリーコンピュータ！

今回作成したコンテンツでは、構造のシンプルなファミコンを用いたため、x86 などのモダンなシステムを解説したものと比べ、アセンブラや物理 RAM などの非常に低レイヤの部分と、その上で実現されるゲームとを、密接に関連させて解説することができる。また、CASL などシンプルだが実在しない題材と比べ、ファミコンは実際に存在するハードウェアであり、高校生でも知っているなど、興味を持ちやすい題材となった。

4. 2　時を司るスクリプト言語「ど～なっつ」

Haskell の継続モナドや Scheme の call/cc などでの「継続」を用いると、ど～なっつでなくとも、それぞれの言語処理系内だけであれば、「時間を遡る」ことが

2　http://donut-lang.org/ ど～なっつ、時を司るスクリプト言語
3　http://donut-lang.org/chisa/ 時を司る GUI：ちさ

可能である。そのことを応用したWebアプリケーションフレームワークKahua [4]なども存在する。しかし、これらの「継続」で戻せるのは処理系の中だけであるため、Webアプリケーションとは違い処理系外への副作用が頻発するGUIアプリケーションでの採用は難しい。

今回、ど～なっつではこの問題を解決するため、「反副作用」という概念を導入した。「反副作用」とは、言語処理系外に起こされる「副作用」を取り消すような副作用である。例えば、「ウインドウを開く」という副作用の反副作用は「ウインドウを閉じる」などの論理的に対になる操作になる。

「ど～なっつ」では各オブジェクトが起こした副作用に対する「反副作用」を保持し、処理系に行われる時間操作に対応した反副作用を適用したり、さらにそこから反・反副作用（≒もとの副作用）を適用することで、時間の矢にそって過去から未来へ任意に外部環境をも移動させることが出来る。

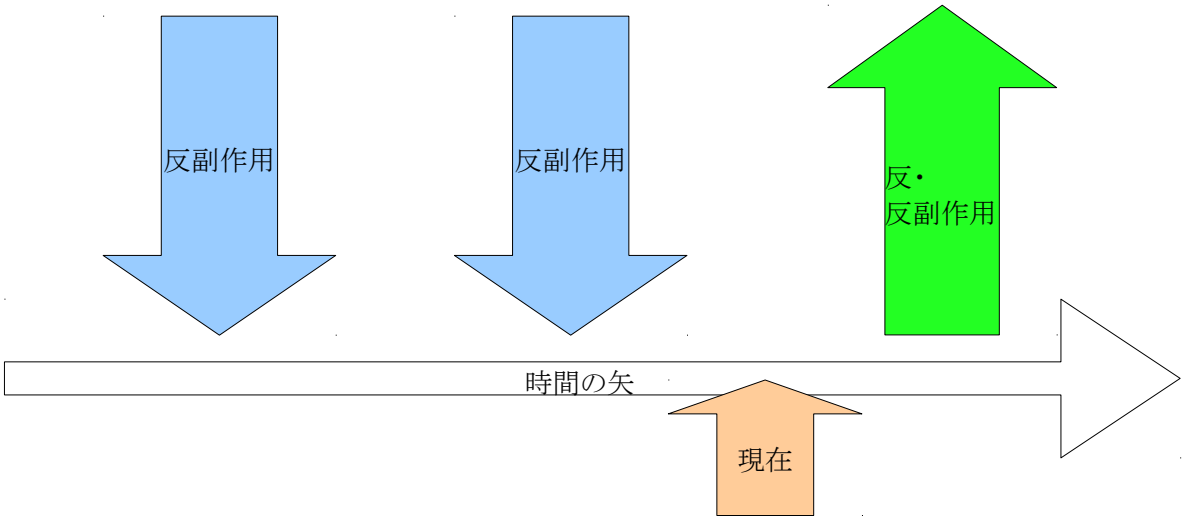

図 2: 一つ時間を巻き戻して反副作用を一つ適用し、さらにその反反副作用を保持している

この「反副作用」のアプローチによって、言語処理系外の環境も包括的に戻すことができる。この性質によって、継続では難しいGUIアプリケーションの時間制御も行うことができ、既存のMementoパターンなどとくらべても開発工数・メモリ消費量を削減することができた。

また、この「時間操作」は言語環境内にある「ほむら」と呼ばれるオブジェクトを使うことで言語環境内からも起こすことができる。この「ほむら」の性質を使うことで、forやwhileなどを用いずに「時間が巻き戻ることを利用したループ」などのフロー制御を作ることができる[5]。

---

4 http://www.kahua.org/ Kahua Project

5 http://donut-lang.org/about/ ど～なっつとは

5. 期待される効果

今回試作したのコンテンツの今後のさらなる発展によって、ハードウェアをブラックボックスと恐れる事無く、「なんだ、蓋を開けてみれば以外と簡単なものの組み合わせなんだな」と思いハードウェアに手を出すエンジニアが増えれば、と期待している。

ど～なっつによって、従来「継続」では副作用のハンドリングが難しかったため実現できなかった、GUI でも「操作をいつでも戻せる」アプリケーションを開発できるようになる。

時間操作で元に戻れるだけでなく、通常のプログラミング言語では作ることのできない「時間操作を用いたフロー制御」なども行えるため、今まで書きづらかったりバグを発生しやすかったりしたユーザーとのインタラクション処理を、もっと簡単に、そして堅牢に書けるようになると期待している。

6. 普及（または活用）の見通し

ワークショップで記述してもらったリアクションペーパーでは、コンテンツへの好意的な評価を得ている。今後はよりコンテンツを拡充し、インターネットの Google Play 上やコミックマーケットなどでの顔を合わせた実地での頒布を考えている。

今後は「ちさ」の教材に限らない GUI フレームワークとしての発展や、「ど～なっつ」の時間の矢を途中で分岐させる、いわゆる「パラレルワールド」への対応を行いたいと考えている。ど～なっつ上での実装で得た知見を、他の言語に移植してもよいかもしれない。

7. クリエータ名（所属）

平藤 燎（東京大学　理科二類）

8. 関連サイト

8. 1　バイナリから目覚めるぼくらのファミリーコンピュータ！

http://ledyba.org/famicom

8. 2　時を司るスクリプト言語「ど～なっつ」　と　時を司る GUI ツールキット「ちさ」

http://donut-lang.org/