狙い通りの機能を実現するために

ロジック回路設計の手ほどき

菅原 孝幸
Takayuki Sugawara

第5回　フリップフロップに入力する信号源を書く

前回(2006年9月号)は，Dフリップフロップのハードウェア記述を行いました．さっそくシミュレーションして動作波形を見てみましたが，入出力とも不定で，動いていませんでした．その理由は信号源がなかったからです．

ハードウェア記述のロジック回路を動作させるには，信号源や測定器にあたるテスト・ベンチが必要です．

今回はそのテスト・ベンチを書いてみましょう．

話を簡単にするために，ハードウェア記述と同じソース・ファイルにテスト・ベンチも書き込んでしまいます．また，測定器にあたる記述は省き，シミュレータの表示機能を利用して，動作を確認します．

クロック源とD入力の信号源を記述する

Dフリップフロップの入力端子は，clockとDの二つです．この両方に信号を用意する必要があります．

● clock端子に入力する発振器を記述してみる

クロックの信号clockを，**リスト5-1**のように記述してみましょう．

clockの初期値を‘0’とします．そのあと，‘0’と‘1’を10単位時間ごとに繰り返します．これで，クロック信号を作ることができます．

シミュレーションは$finishがないと止まりません．1000単位時間後に終了させます．

clock信号の記述とシミュレーション終了の記述，この二つはテスト・ベンチということになります．

これらにD型フリップフロップのハードウェア記述を加えたソース・ファイルが**リスト5-2**です．

リスト5-2をシミュレーションすると，**図5-1**のように四つのプロセスが並走します．

▶ 波形表示までのおさらい

リスト5-1のファイルをdff_test2.vとして保存します．保存したファイルをVeritakから開き，[Go]ボタンを押してシミュレーションします．

[ScopeTreeView]ボタン([Go]ボタンの五つ右)を押すと，信号名が表示された画面が現れます．表示されたD，Q，clockの三つの信号すべてをドラッグして選択したあと，右クリックでメニューを表示させ，

リスト5-1　clock源を記述する

```
initial begin
    clock=0;
end

always begin
    #10;
    clock=~clock;
end
```

clockの初期値を‘0’に設定

10単位時間ごとにclockを反転する

Keyword 1　セットアップ・タイム/ホールド・タイム

今回のDフリップフロップでは，クロックの立ち上がりエッジでD端子の入力信号を読み込みます．

このとき，クロックの立ち上がりよりも前に，Dの値を決めて保持しておかないと，Qの値がどうなるか確実ではありません．

最小限，Dを保持しておかなければいけない時間を，セットアップ・タイムといいます(**図5-A**)．

同様に，クロックが立ち上がった後も，入力Dをある程度の時間，保持しておかなければいけません．

この，立ち上がり後にDを保持しておかなければいけない最小限の時間を，ホールド・タイムといいます(**図5-B**)．

セットアップ・タイムやホールド・タイムは，そのデバイスのデータシートを見ると書いてあります．

例えば，セットアップ・タイムが3 ns，ホールド・タイムが2 nsであれば，D入力はクロックの立ち上がりの3 ns以上前に値を確定させ，さらにクロックの立ち上がり後も2 ns以上はずっと同じ値を保持しておく必要があります．

以上の話は，立ち上がりエッジのD型フリップフロップですが，立ち下がりでも同様です．

このセットアップ・タイムとホールド・タイムの両方を

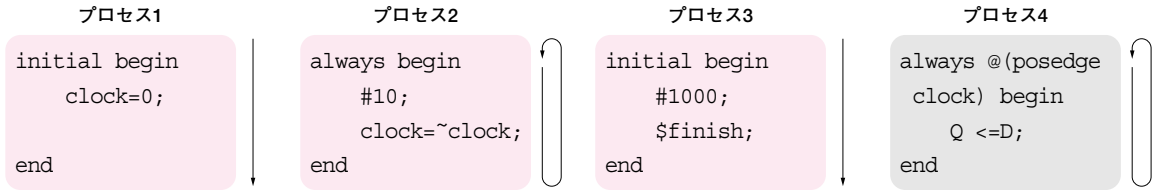

図5-1　リスト5-2は四つのプロセスが同時に処理される
initial文は1回だけ，always文は何度も処理される

図5-2　Dフリップフロップのclock端子に信号が入ったのがわかる
リスト5-2のシミュレーション結果

［選択信号をWaveformViewerに追加］を選びます．

するとWaveform Viewerの画面が出てきます．左端の［Entire Wave］ボタンを押すと，波形全体が表示されます．その右隣の［+］，［−］ボタンを押すと，時間軸方向に拡大縮小ができます．

適当なサイズで波形を表示させたのが**図5-2**です．

● D端子に入力する信号源を記述してみる

図5-2を見ればわかるように，まだDを動かしていないので，DもQも不定のままです．

そこで，**リスト5-3**のようにDに関する記述を追加します．初期値は‘0’で，20単位時間後に‘1’になり，その20単位時間後（最初から40単位時間後）に‘0’になります．その20単位時間後（最初から60単位時間後）に‘1’になると，そのあとは指定がないので，シミュレーション終了まで‘1’のままです．

リスト5-3のシミュレーション結果は**図5-3**です．

期待どおり，clockの立ち上がりで，D入力の値を読み込んでいるようすがわかります．

リスト5-2　clockの記述とシミュレーション終了の記述を加えたDフリップフロップ

```
module dff_test2;
    reg D,Q;
    reg clock;

    initial begin
     clock=0;
    end                    プロセス1

    always begin
     #10 ;
     clock=~clock;
    end                    プロセス2

    initial begin
     #1000;
     $finish;
    end                    プロセス3

    always @(posedge clock) begin
     Q <=D;
    end                    プロセス4

endmodule
```

リスト5-1のクロック信号
1000単位時間後にシミュレーション終了
テスト・ベンチ
Dフリップフロップのハードウェア記述

これで，Dフリップフロップというハードウェア記述の動作確認が終わりました．

クロックとデータが同時に変化する記述はNG

● clockとDを同時に動作させてみる

ところで，この**リスト5-3**のDフリップフロップのシミュレーションで，clockとDを同時に変化させると，どうなるのでしょうか？

Keyword 1　セットアップ・タイム/ホールド・タイム（つづき）

満足すれば，出力が保証されます．

これを満足しないときは，出力が保証されない，つまり，‘0’になるか‘1’になるかわかりません．

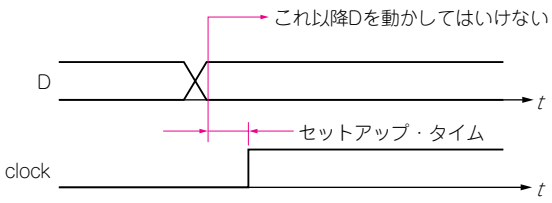

図5-A　セットアップ・タイムの定義

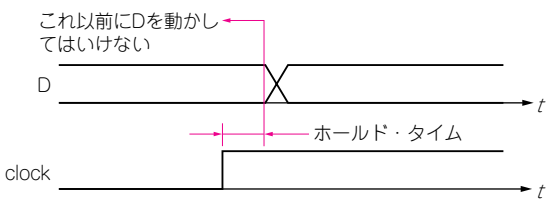

図5-B　ホールド・タイムの定義