第2部　第2章

記述力の高いChiselを使って

UDPを実装してIP通信する

井田 健太

リスト1　モジュールの定義

```
class MyModule extends Module {
  val io = IO(new Bundle {
    val in_a = Input(Bool())
                        // Input(...) で入力ポートを定義
    val in_b = Input(Bool())
    val out = Output(Bool())
                        // Output(...) で出力ポートを定義
  })
  // モジュールの論理回路を書く
}
```

イーサネットMACを実装したので，次に上位プロトコルの対応をします．ホストPCなどとイーサネットで通信するために必要な各種プロトコルの処理も論理回路に実装します．これらの論理回路を実装するにあたって，イーサネットMACの実装に用いたSystemVerilogではなく，Chiselという別のハードウェア記述言語を用います．

● Chiselは強力な記述機能を持ったHDL

Chiselは，Verilog HDLやVHDLと同じくハードウェア記述言語（HDL）の一種です．Java仮想マシン（JVM）上で動作するプログラミング言語Scalaの内部DSL（Domain Specific Language），つまりScalaのサブセット言語として実装されており，Scalaの強力な型システムや記述力を用いながら，論理回路を記述できます．

言語自体がScalaのサブセットであるため，Scalaに対応した開発環境が利用できます．筆者はVisual Studio Codeと，そのScala向け拡張機能であるMetalsを組み合わせて開発しています．

Scalaはソフトウェア開発を主目的とした言語ですので，他のソフトウェア開発用の言語と同じように，構文補完やリファクタリング，テスト・フレームワークといった機能が利用できます．

リスト3　switch文の記法

```
switch(式) {
  is( 値1 ) {

  }
  is( 値2 ) {

  }
}
```

リスト2　if文の記法

```
when( 条件1 ) {
  // 条件1 で実行する処理
} .elsewhen( 条件2 ) {
  // !条件1 && 条件2 で実行される処理
} .otherwise {
  // いずれの条件も満たさない場合の処理
}
```

Chiselはいわゆる高位合成言語と間違われることが多いですが，ハードウェア記述言語としての抽象度はSystemVerilogと同じです．

Chisel記法の簡単な説明

Scalaの文法は知っていることを前提に，Chiselで使う型などをSystemVerilogと比較しつつ説明します．

▶モジュールの記述

Chiselでは，`Module`クラスを継承した型としてモジュールを表現します．ユーザが作る型の中で`io`という名前のフィールドを定義することでポートを表現します．

`MyModule`という名前のモジュールを定義する例を**リスト1**に示します．`MyModule`は入力ポートとして，`in_a`，`in_b`，出力ポートとして`out`を持ちます．

▶条件分岐

SystemVerilogでは`if`文や`case`文で条件分岐を表します．Chiselでは`when`文が，SystemVerilogの`if`文に対応します（**リスト2**）．

`.elsewhen`や`.otherwise`ブロックは省略可能です．

SystemVerilogの`case`文に対応するのが`switch`文です（**リスト3**）．`switch`に指定した式の値が，`is`に指定した値に対応する場合，`is`のブロック内の処理が実行されます．SystemVerilogの`case`文の`default`節に対応する機能はありません．

▶関数定義

SystemVerilogの`function`文と同じように，Chiselでは Scalaの`def`文を使って関数を定義できます（**リスト4**）．