

Ship Constants

Adam Megacz

August 7, 2007

Abstract

This memo explains how the Fleet assembler syntax deals with named constant values. This is strictly a matter of syntax and parsing, and has no impact on the actual silicon implementation.

1 Constants

A *constant* is a named value or bit mask. A constant can be associated with a particular port on a ship (a *port constant*) or can be associated with all ports on the entire ship (a *ship constant*).

Constants are declared in the `.ship` file for a ship. Ship constants are declared in the `== Constants ==` section; port constants are declared in the `== Ports ==` section immediately after the port to which they belong.

2 Anatomy

Every constant is a bit mask; that is, it is an array of bits whose width is equal to the word width of the machine. Each bit in this mask can be of six types, each designated by a character

- “0” means the bit is cleared
- “1” means the bit is set
- “.” means the bit is unchanged
- “s” means the bit is set to a sign-extended constant¹
- “u” means the bit is set to a non-sign-extended (“unsigned”) constant

A given constant may not contain both `s` and `u` bits. Moreover, all `s` or `u` bits must be contiguous.

A constant may be specified using either decimal, hexadecimal, or binary. Hexadecimal constants are distinguished by a preceding `0x`. Only binary constants may have bits of the last three types of bits – this means that constants specified as decimal or hexadecimal values are full-word-width constants.

A constant is specified by the word `constant` followed by the name of the constant, followed by a colon, followed by the constant itself. All must appear on the same line. Binary numbers are written in *big-endian* format.

¹that is, whatever value the programmer provides will be sign-extended to fill the bits allotted

2.1 Examples

Here are some examples of constants:

```

== Constants =====
constant FiftyTwo: 52
constant FiftyThree: 0x35

== Ports =====
port inDataOp
  constant rev: .....1.....
  constant inv: .....1.....
  constant count: .....uuuuu.....
  constant offset: .....sssss

```

3 Using Constants

Constants may be used in Fleet syntax wherever a literal value (such as 12) would be permitted.

A constant must be specified in the form `Ship[CONSTLIST]` or `Ship.Port[CONSTLIST]` where:

- `Ship` is the name of a ship *or a ship instance*.²
- `Port` is the name of a port on the ship.
- `CONSTLIST` is a comma-separated list of one or more constants. Each constant in the list is the name of a constant and optionally the character = followed by a decimal number, hexadecimal number, or the name of another constant.

3.1 Special Exceptions

The `Ship` and `Port` elements may be omitted when a constant is being used as an opcode, and when the ship and port match the ship and opcode port which are targets of the instruction.

²If a ship and some instance have the same name, the instance is preferred.

3.2 Examples

```
MyShip myship;

myship[FiftyTwo]:          sendto myship.inDataOp;
myship.inDataOp[rev]:     sendto myship.inDataOp;
myship.inDataOp[count=3]: sendto myship.inDataOp;
myship.inDataOp[count=3,inv]: sendto myship.inDataOp;
myship.inDataOp[count=inv]: sendto myship.inDataOp;

// the following two do the same thing
72: sendto myship.inData(myship.inDataOp[count=inv]);
72: sendto myship.inData(          [count=inv]);
```