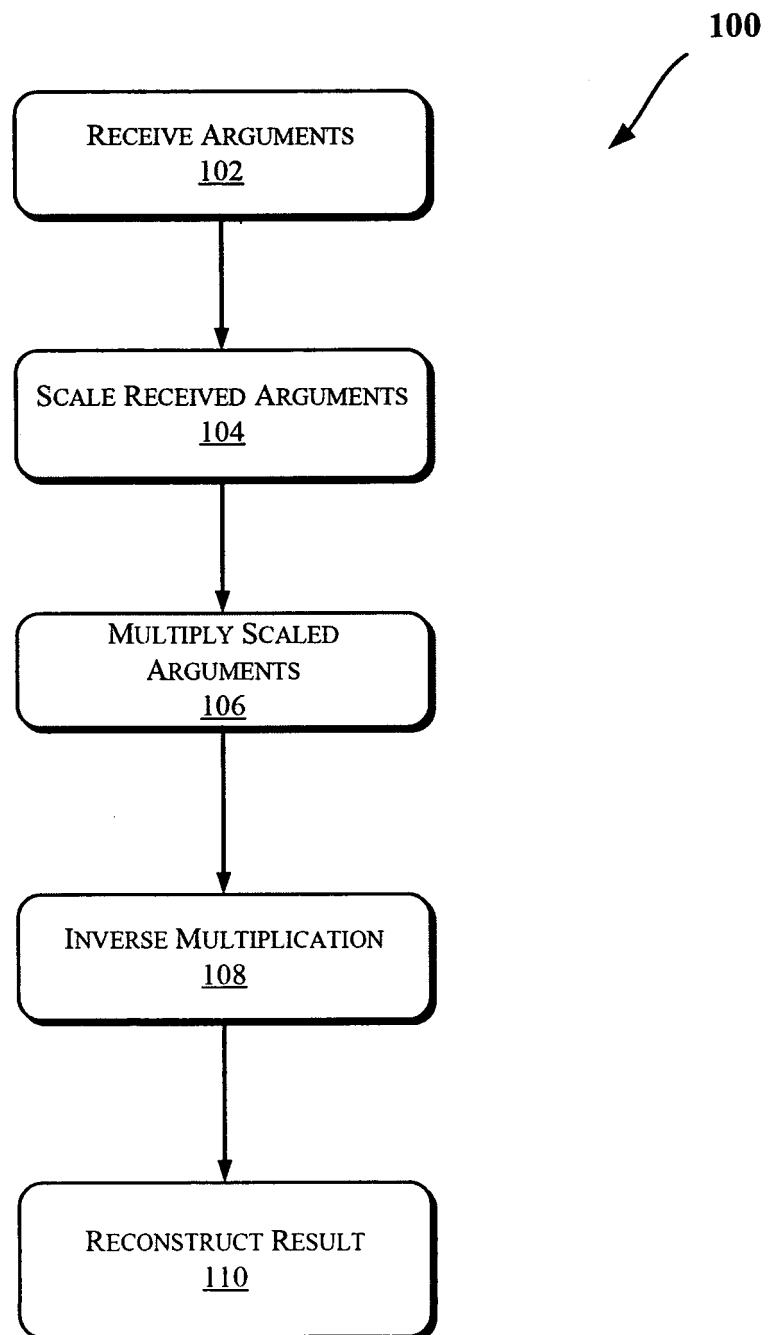


*FIG. 1*

```

#-----
# a[1],a[2],a[3],a[4] – arguments
# r[1],r[2],r[3],r[4] – results
# F2I – floating point to integer value cast
# I2F – integer to floating point value cast
# SIGN_AND_MANTISSA_MASK – integer mask with zeros
# in exponent field. In case of IEEE double precision
# SIGN_AND_MANTISSA_MASK = 0x800FFFFFFFFFFFFFFF
#=====

```

FIG. 2A

```

# Scaling of arguments by exponent of 1.0
# Saving original exponent values as integers
for i from 1 to 4 by 1 do
    a_scaled[i] := I2F(F2I(a[i]) & SIGN_AND_MANTISSA_MASK) | F2I(1.0);
    i_a_exp[i] := F2I(a[i]) & ~(SIGN_AND_MANTISSA_MASK);
end do;

```

```

# Scaled arguments multiplication and single bundle-division
R := 1/( a_scaled[1] * a_scaled[2] * a_scaled[3] * a_scaled[4]);

```

```

# Scaled results computation
r_scaled[1] := R * a_scaled[2] * a_scaled[3] * a_scaled[4];
r_scaled[2] := R * a_scaled[1] * a_scaled[3] * a_scaled[4];
r_scaled[3] := R * a_scaled[1] * a_scaled[2] * a_scaled[4];
r_scaled[4] := R * a_scaled[1] * a_scaled[2] * a_scaled[3];

```

FIG. 2B

```

# Results reconstruction by negated source exponents
for i from 1 to 4 by 1 do
    r[i] := I2F(F2I(r_int[i]) + F2I(1.0) – i_a_exp[i]);
end do;

```

FIG. 2C

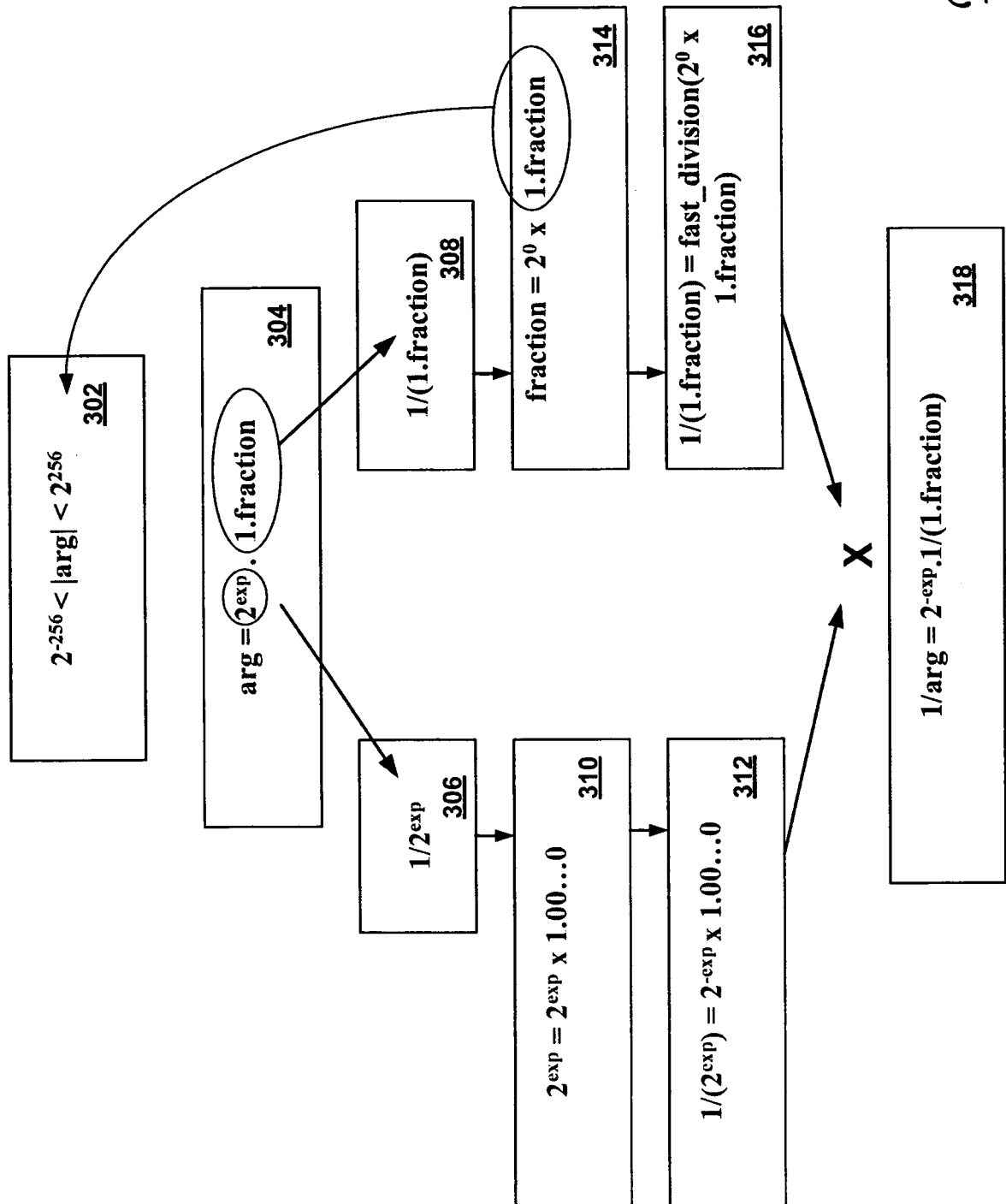
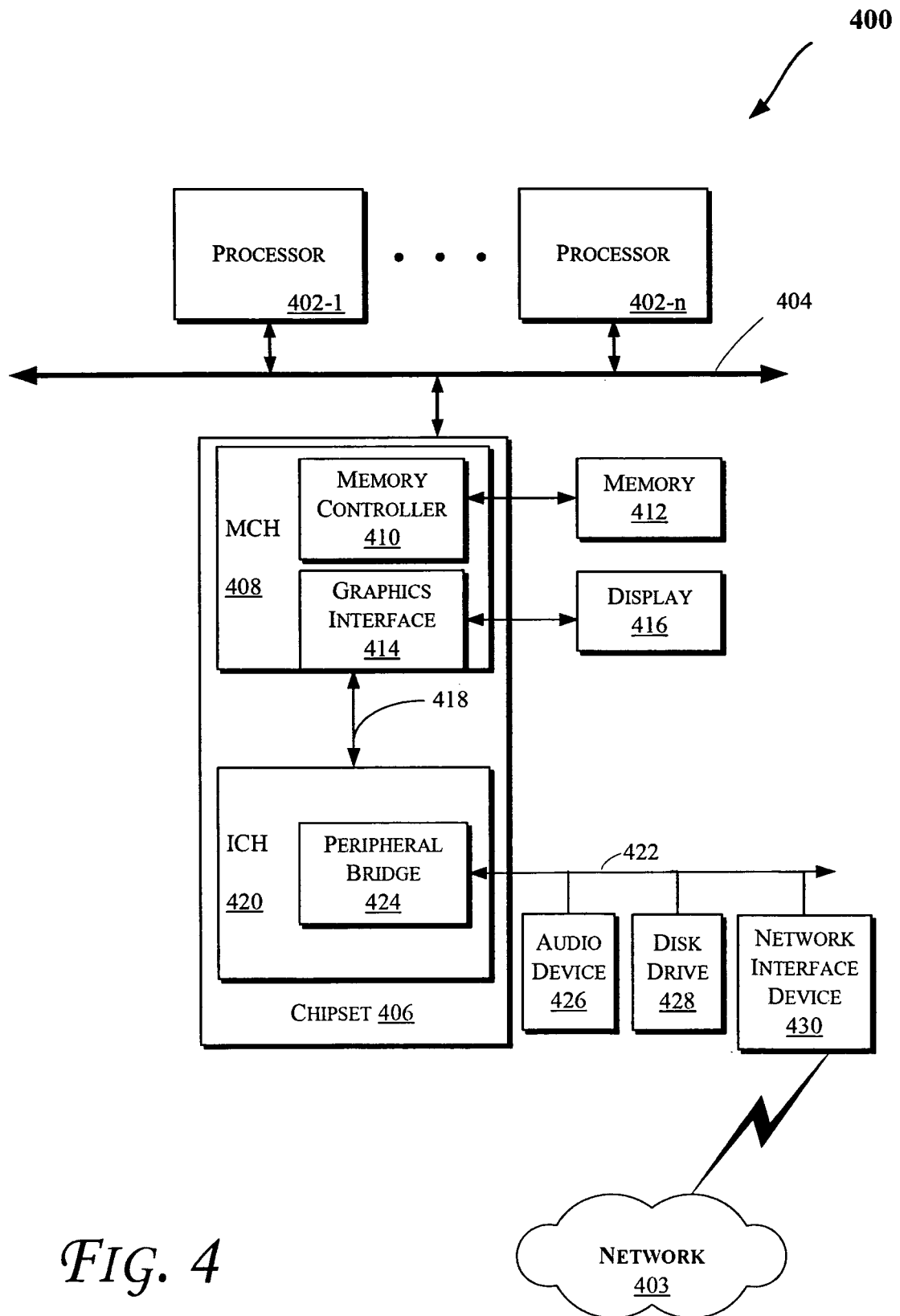


FIG. 3



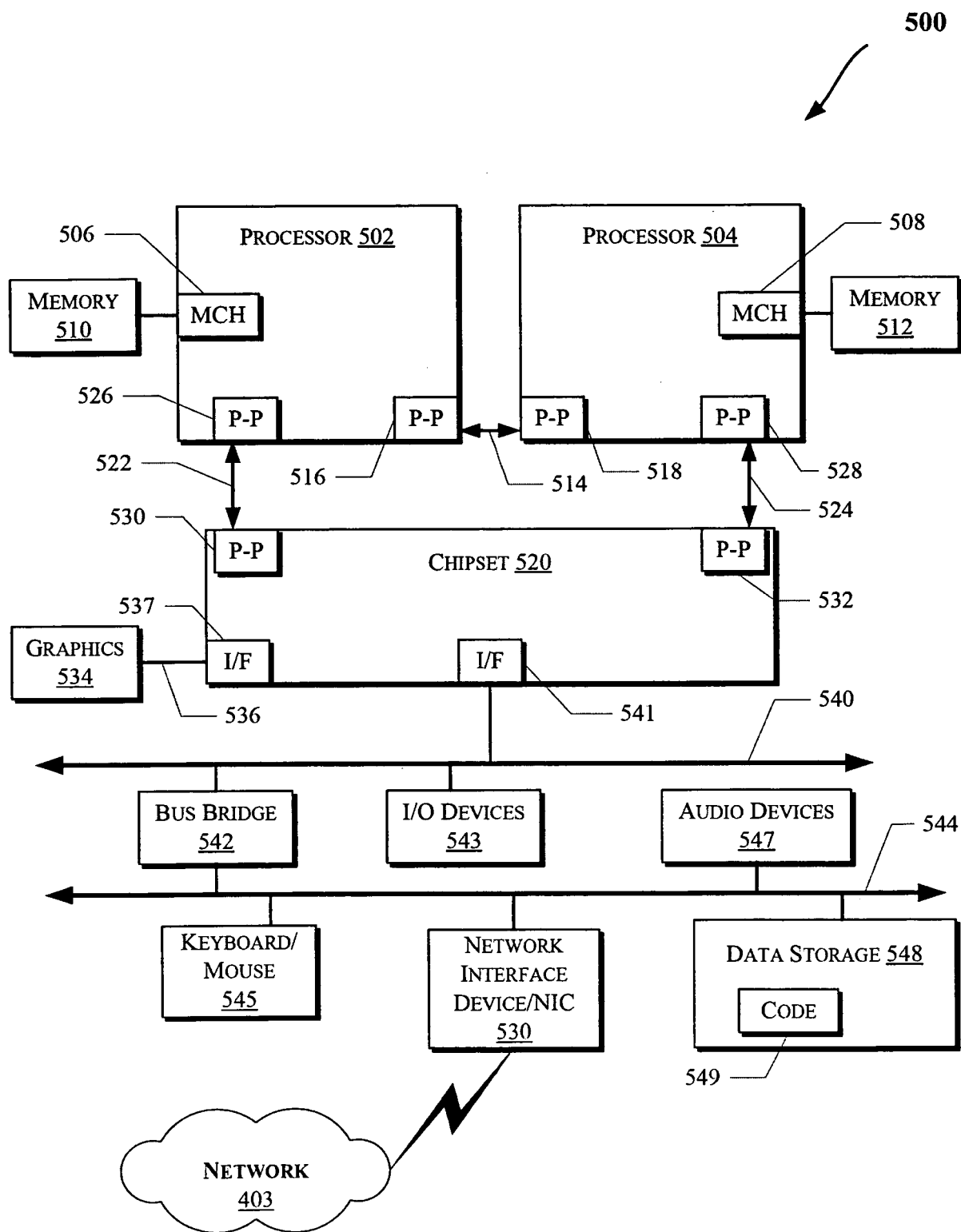


FIG. 5