**Arduino Due (**Atmel SAM3X8E **84 MHz** Cortex-M3**)**  
INT\_LOOP(30000) bench...= 1074 microseconds 27.93MIPS  
LONG\_LOOP(30000) bench...= 1107 microseconds 27.10MIPS  
FLOAT\_DIV(30000) bench...= 25859 microseconds 1.21MFLOPS  
DOUBLE\_DIV(30000) bench...= 37966 microseconds 0.81MFLOPS  
FLOAT\_MUL(30000) bench...= 18659 microseconds 1.71MFLOPS  
DOUBLE\_MUL(30000) bench...= 25450 microseconds 1.23MFLOPS

**Arduino Due (**Atmel SAM3X8E **84 MHz** Cortex-M3**) -**

16:30:50.961 -> INT\_LOOP(30000) bench...= 1075 microseconds 27.91MIPS

16:30:50.961 -> LONG\_LOOP(30000) bench...= 1108 microseconds 27.08MIPS

16:30:50.994 -> FLOAT\_DIV(30000) bench...= 25500 microseconds 1.23MFLOPS

16:30:50.994 -> DOUBLE\_DIV(30000) bench...= 1108 microseconds infMFLOPS

16:30:51.028 -> FLOAT\_MUL(30000) bench...= 20121 microseconds 1.58MFLOPS

16:30:51.028 -> DOUBLE\_MUL(30000) bench...= 1106 microseconds -15000.00MFLOPS

**Teensy 3.2 (**MK20DX256Cortex-M4 **72MHz)**  
INT\_LOOP(30000) bench...= 1253 microseconds 23.94MIPS  
LONG\_LOOP(30000) bench...= 1256 microseconds 23.89MIPS  
FLOAT\_DIV(30000) bench...= 14635 microseconds 2.24MFLOPS  
DOUBLE\_DIV(30000) bench...= 25083 microseconds 1.26MFLOPS  
FLOAT\_MUL(30000) bench...= 11288 microseconds 2.99MFLOPS  
DOUBLE\_MUL(30000) bench...= 17551 microseconds 1.84MFLOPS

**Arduino Nano (**ATMega328 **16MHz AVR)**  
INT\_LOOP(30000) bench...= 7544 microseconds 3.98MIPS  
LONG\_LOOP(30000) bench...= 13408 microseconds 2.24MIPS  
FLOAT\_DIV(30000) bench...= 154792 microseconds 0.21MFLOPS  
DOUBLE\_DIV(30000) bench...= 154800 microseconds 0.21MFLOPS  
FLOAT\_MUL(30000) bench...= 156744 microseconds 0.21MFLOPS  
DOUBLE\_MUL(30000) bench...= 156736 microseconds 0.21MFLOPS

**Xirka Ardunesia (XST X1-02 Cortex-M3 50MHz)**

15:45:20.845 -> Time (ms)...= 11737 ms

15:45:20.845 -> INT\_LOOP(30000) bench...= 1801 microseconds 16.66MIPS

15:45:20.845 -> LONG\_LOOP(30000) bench...= 1889 microseconds 15.88MIPS

15:45:20.878 -> FLOAT\_DIV(30000) bench...= 21161 microseconds 1.56MFLOPS

15:45:20.878 -> DOUBLE\_DIV(30000) bench...= 1889 microseconds infMFLOPS

15:45:20.911 -> FLOAT\_MUL(30000) bench...= 16323 microseconds 2.08MFLOPS

15:45:20.911 -> DOUBLE\_MUL(30000) bench...= 1887 microseconds -15000.00MFLOPS

**Arduino Zero** (Atmel ATSAMD21G18 **48MHz** Cortex-M0+)  
INT\_LOOP(30000) bench...= 116898 microseconds 11.92MIPS  
LONG\_LOOP(30000) bench...= 116898 microseconds 11.93MIPS  
FLOAT\_DIV(30000) bench...= 116898 microseconds 0.38MFLOPS  
DOUBLE\_DIV(30000) bench...= 113126 microseconds 0.27MFLOPS  
FLOAT\_MUL(30000) bench...= 92387 microseconds 0.33MFLOPS  
DOUBLE\_MUL(30000) bench...= 116898 microseconds 0.26MFLOPS

**Teensy LC** (MKL26Z64 Cortex-M0 **48MHz**)  
INT\_LOOP(30000) bench...= 2508 microseconds 11.96MIPS  
LONG\_LOOP(30000) bench...= 2512 microseconds 11.94MIPS  
FLOAT\_DIV(30000) bench...= 76705 microseconds 0.40MFLOPS  
DOUBLE\_DIV(30000) bench...= 101840 microseconds 0.30MFLOPS  
FLOAT\_MUL(30000) bench...= 80471 microseconds 0.38MFLOPS  
DOUBLE\_MUL(30000) bench...= 106242 microseconds 0.29MFLOPS

How MIPS is calculated?

Alternatively, divide the number of cycles per second (CPU) by the number of cycles per instruction (CPI) and then divide by 1 million to find the **MIPS**. For instance, if a computer with a CPU of 600 megahertz had a CPI of 3: 600/3 = 200; 200/1 million = 0.0002 **MIPS**.