

1. In Russell Massey 's article, interrupts are defined as events outside the running program. Interrupts can separate non-time-critical functions from time-critical functions in a program. Time-critical functions are executed on response to interrupts. Most CPUs and devices can either enable or disable interrupts with some mechanisms. Most current CPU uses an interrupt vector table to deal with interrupts. Many interrupts can provide priorities among different interrupts sources. That is, when multiple interrupts happen simultaneously, time-critical interrupts are processed prior to non-time-critical interrupts. Regarding the interrupt latency, high-priority interrupts also experience lower delay in processing.

2. Interrupt latency is defined as the time between when the interrupt occurs and when the CPU suspends its current job. This article shows most of the latency is due to software design. In ISR, user needs to enable interrupt as soon as possible to reduce latency. The interrupt latency is easy to measure with some codes in ISR.

3. Since interrupts are considered as simultaneous if they occur between the same two clock ticks, the time should be $\text{Time} = 1/8\text{MHz} = 0.125 \times 10^{-6}$ seconds

4. The interrupts are **Reset, NMI, Watchdog Timer, Single-Step, Timer B2, ADC Conversion Complete, UART0 Receive, UART2 Transmit** in the order of decreasing priorities.

5.

Address	Value
0x07f9	-
0x07fa	-
0x07fb	-
0x07fc	12
0x07fd	F3
0x07fe	20
0x07ff	3f
0x0800	-
0x0801	-

Register	Initial Value	Final Value
SP	0x0800	0x07fc
PC	0x0ff312	0x0ff440
FLG	0x3062	0x3020