Introduction to Interrupts
First Form of Parallelism

Dominique Thiébaut
dthiebaut@smith.edu
References

• See class page for references:

• http://www.science.smith.edu/dftwiki/index.php/CSC352_Class_Page_2017
Simplified view of Computation
Simplified view of Computation
Example Program 1

This is the text for an editor that gets characters from the keyboard, and saves and closes the program on Ctrl-X.

```c
...
init();
...
while (true) {
    while (!has_char()) {
        ch = get_char();
        if (ch == ^X) {
            savefile();
            exit (0);
        } else
            ...
} else
    ...
} else
    if ('z' >= ch && 'a' <= ch) {
        insertChar(ch);
    }
}
```
Example Program 2

This is the same text for an editor, but more contemporary. What's different?

... CreateWindow(); EnableEvent(WM_CLOSE); ...

void eventOccurred(Event e) {
    switch (e.code) {
        case WM_CLOSE:
            savefile();
            exit(0);
        case 'a'-'z':
            insertChar(e.code);
            break;
        default:
            break;
    }
}
The Reality
The Reality

Time

Time

Time
How do Interrupts Work?

• Hardware
• Processor
• Stack
Infrastructure

Processor

I/O Controllers

Ram
Response to an Interrupt

• At every new instruction:
  
  • if interrupt pending and interrupts allowed…
How fast?

• How fast is a context switch, approximately?
What's a more accurate graph?
Quantum

- The operating system typically allows programs/processes to run for a fixed amount of time before another process takes over the processor. How can this be implemented?
That's the root of Parallelism!

Image credits:
Process vs Threads
Goals of Multithreading

• Enhance performance
• Increase throughput
• Divide the work into well defined tasks that can be idle waiting for information
• Greater user responsiveness
Caveat

• Python supports multithreading, and multiprocessing.

• Python threads CANNOT RUN IN PARALLEL (GIL)

• If parallelism is needed in Python, use the Multiprocessing library

Examples

• Go to class Web page: http://www.science.smith.edu/dftwiki/index.php/Python_Multithreading/Multiprocessing_Examples
Monte-Carlo Pi

\[
m = 281 \\
n = 233 \\
\Pi = 4 \times \frac{n}{m} \\
\Pi = 3.316725978647687
\]

http://montepie.herokuapp.com/
Python

```python
def monteCarloPi():
    N = 1000000  # int( input( "> " ) )
    inside = 0
    for i in range( N ):
        x = random()
        y = random()
        if x**2 + y**2 < 1:
            inside += 1

        if i > 0 and i%10000 == 0:
            print( "%9d %1.12f" % ( i, 4.0*inside/i ) )
```
Write a multiprocessing application in Python that computes an approximation of Pi using the Monte Carlo simulation, and using 10 Processes.