

ERROR HANDLING IN CONCURRENT ERLANG

Curt Clifton
Rose-Hulman Institute of Technology

SVN Update *ErlangInClass/lifts*

GETTING DOWN WITH LIFTS

- Time-limits for *receive*
- Multiple processes

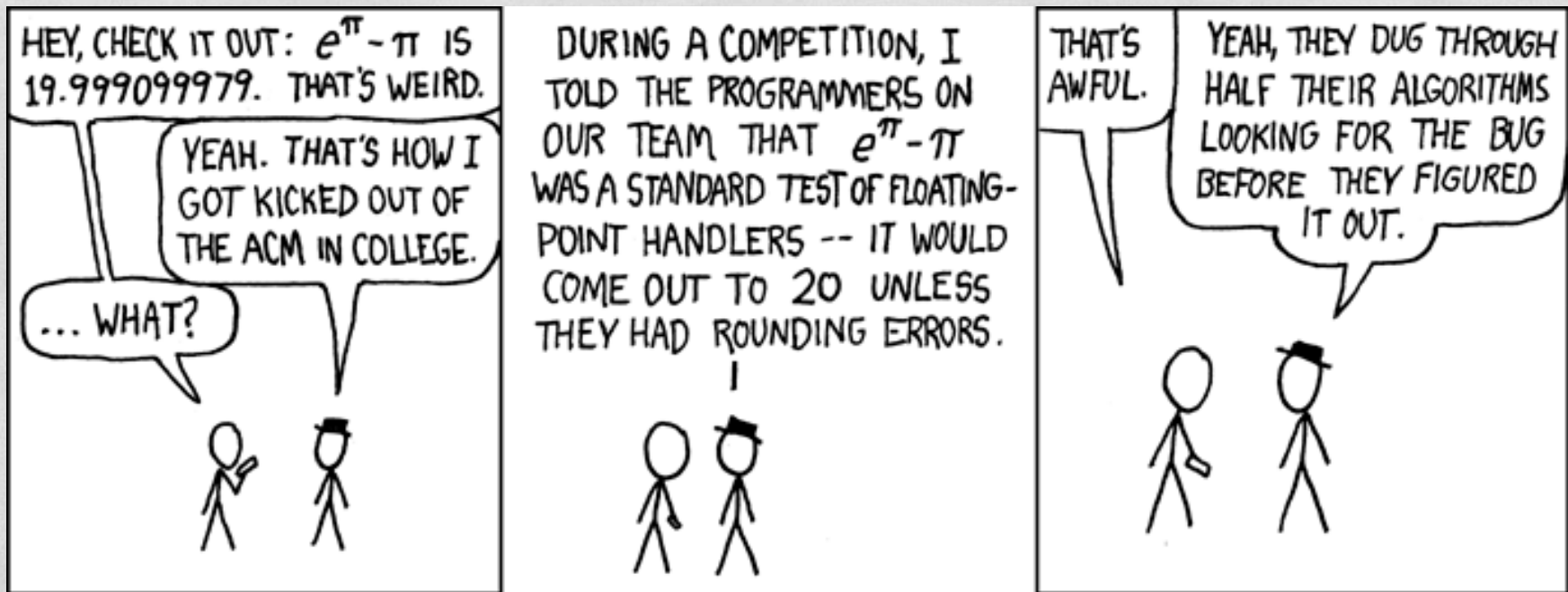
USING TIME-OUTS TO CREATE PERIODIC EVENTS

- Open *lifts_v3.erl*
- Notice:
 - *monitor_car* function
 - *car_hardware* and *car_loop* interaction

REGISTERED PROCESSES

- Some processes are intended to handle messages from throughout the system
- Can *register* these so we don't have to pass around their PIDs
- Four BIFs for this:
 - *register(AnAtom, Pid), unregister(AnAtom)*
 - *whereis(AnAtom), registered()*

E TO PI MINUS PI



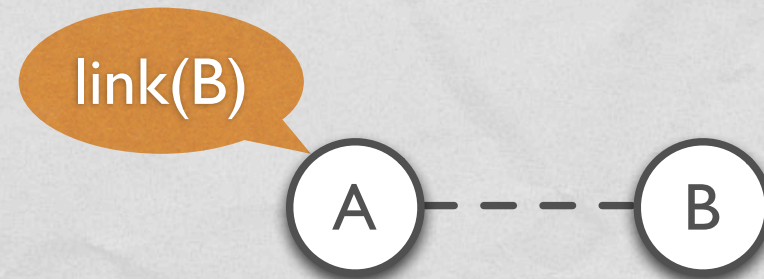
Also, I hear that the fourth root of $(9^2 + 19^2/22)$ is π

CONCURRENT ERROR HANDLING

- Relies on:
 - Linked processes
 - Exit signals
 - System processes

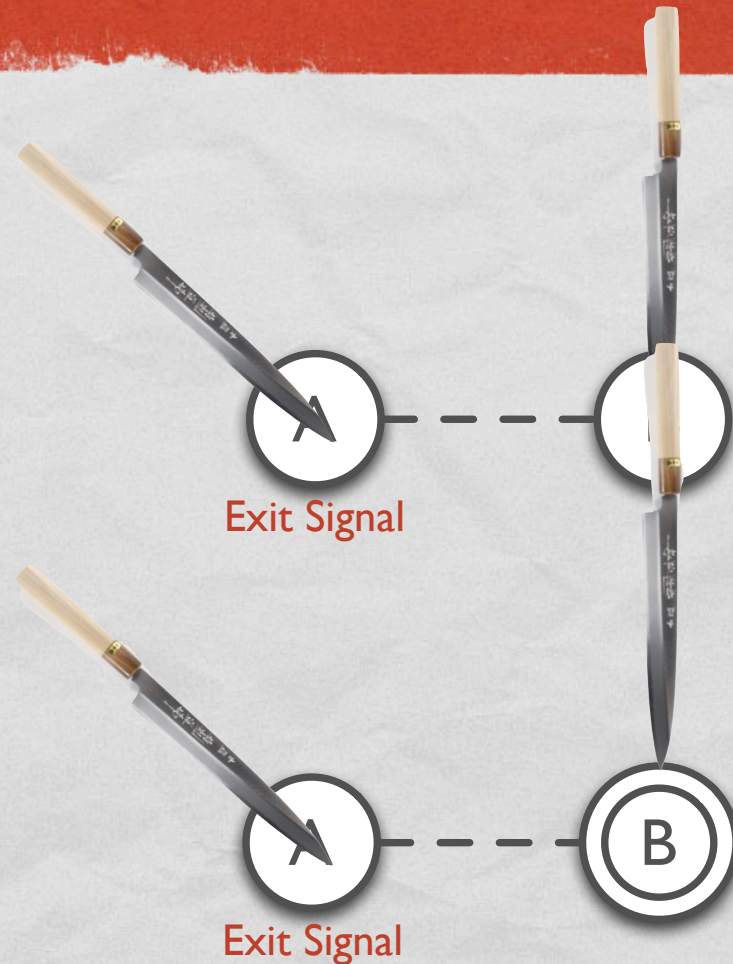
FORMING LINKS

- Any process can link with another using *link(Pid)*
- Link is symmetric
- If either dies, the other receives an *exit signal*

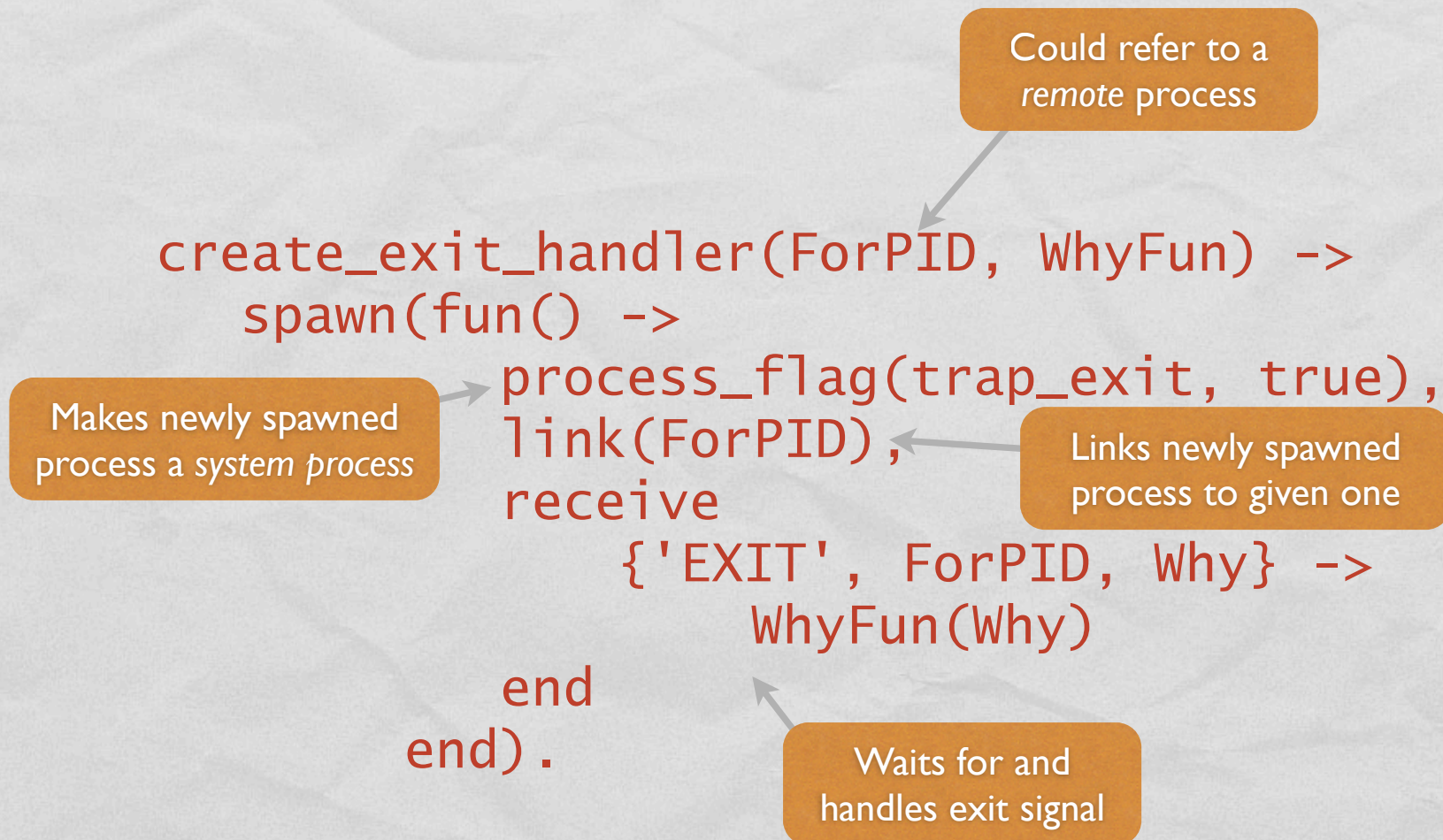


EFFECTS OF EXIT SIGNALS

- Regular process:
 - Dies too
- System process:
 - Traps signal and handles it



SYSTEM PROCESS EXAMPLE



LINK SETS

- A process **P** can be linked to several other processes
- Those processes are called the *link set* of **P**
- An exit signal generated by **P** is broadcast to all processes in **P**'s link set

GENERATING EXIT SIGNALS

- Explicitly: *exit(Reason)*
- Implicitly: when uncaught error occurs
- Normally: when process runs off the end
- For insurance purposes: *exit(Pid2, Reason)*

HANDLING DIFFERENT EXITS SIGNALS

Exit Signal Received	<i>trap_exit</i> Setting	Action Taken
<i>kill</i>	<i>true</i>	Die, broadcasting <i>killed</i> to link set
<i>killed</i>	<i>true</i>	Continue, adding {'EXIT', <i>Pid</i> , <i>killed</i> } to mailbox
<i>Msg</i>	<i>true</i>	Continue, adding {'EXIT', <i>Pid</i> , <i>Msg</i> } to mailbox
<i>normal</i>	<i>false</i>	Continue, ignoring the signal
<i>kill</i>	<i>false</i>	Die, broadcasting <i>killed</i> to link set
<i>Msg</i>	<i>false</i>	Die, broadcasting <i>Msg</i> to link set

System Processes

Normal Processes

EXIT TRAPPING IDIOMS

- Heartless: I don't care if you die
 - Just *spawn* new process
- Romeo: I want to die if you die
 - Don't trap exits
 - Use *spawn_link* to simultaneously spawn and link to new process
- Executor: I'll handle your affairs if you die
 - Trap exits and use *spawn_link* for new process