`amforth`: multitasking

Erich Wälde

Forth Gesellschaft e.V. 2012

Task Control Block

- state (sleep or awake)
- follower (tid of next task)
- `rp0` base of return stack
- `sp0` base of data stack
- `sp` top of data stack
- catch/throw handler
- base
- deferred words: `emit` …

Task Switch

- store relevant info of current task into TCB
- lookup next task
- switch user area pointer to new TCB
- restore relevant info of new task into stacks and pointers
- resume execution of new task

Collaborative multitasking: All tasks need to give up control regularly by calling `pause`

Example

- task 1:
  - command loop
- task 2:
  - increment N
  - store N on Port B
  - wait 500 ms

talk to Port B

```
$38 constant PORTB
$37 constant DDRB

variable N
: init
  $ff PORTB c!     \ portB: all pins high
  $ff DDRB  c!     \        all pins output
  0 N !
;
```

load multitasker

```
include multitask.frt
```

code of task 2

```
\ call pause on wait
: ms ( n -- ) 0 ?do pause 1ms loop ;

: run-demo         \ --- task 2 ---
  begin
    N @ invert PORTB c!
    1 N +!
    &500 ms
  again
;
```

create task 2

```
\ d-stack-size
\ r-stack-size
\ additional-user-area-size
\ --
$20 $20 $10 task: task_demo

: start-demo
  task_demo tcb>tid activate
  \ remainder runs in new task
  run-demo
;
```

setup and start

```
: starttasker
  task_demo task-init
  start-demo

  onlytask
  task_demo tcb>tid alsotask

  multi
;
```

run the show

```
: run-turnkey
  applturnkey
  init
  starttasker
;

' run-turnkey is turnkey
```

live

```
> hex tasks
93   running
135   running
Multitasker is running ok
> N @ .
148B  ok
> 0 N !
 ok
> N @ .
9  ok
>
```

Thank You!

slide set generated with $\LaTeX$/Beamer(Manhattan.sty)