Mittuniversitetet
MID SWEDEN UNIVERSITY

ET014A/ET016A: Sensor Networks

Sensor Node Programming

Sebastian Bader

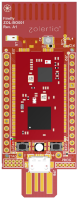
Department of Computer and Electrical Engineering

Operating Systems

- Interface between hardware and software
- Makes resources more convenient to be used (simplification)
- Provides efficient and fair allocation of resources

Operating Systems in Embedded Systems?

VS



Multi-threaded programming model

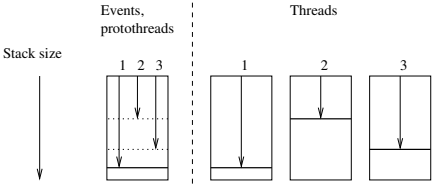
- Multiple (independent) threads run in parallel
- Each thread requires their own resources
- OS is responsible for scheduling

Event-driven programming model

- Typically only one single application
- Program flow is determined by events
- Common approach in embedded systems

Protothreads

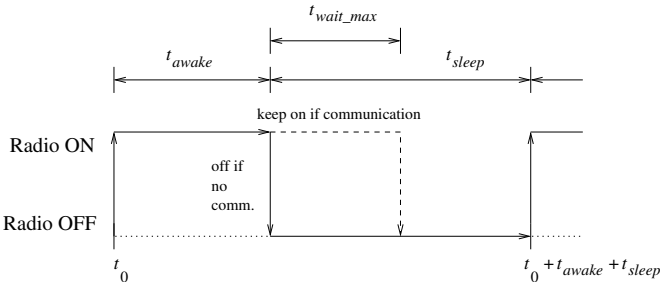
- Combines the advantages of multi-threaded and event-driven programming models



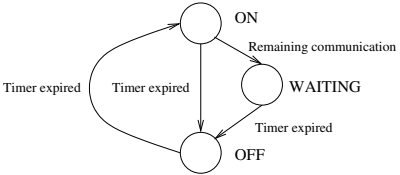
Hello World with Protothreads

```c
#include "contiki.h"

#include <stdio.h> /* For printf() */

PROCESS(hello_world_process, "Hello world process");
AUTOSTART_PROCESSES(&hello_world_process);

PROCESS_THREAD(hello_world_process, ev, data)
{
  PROCESS_BEGIN();

  printf("Hello, world\n");

  PROCESS_END();
}
```

Protothread example



Example: State machine implementation



state: {*ON*, *WAITING*, *OFF*}

```
radio_wake_eventhandler:
  if (state = ON)
    if (expired(timer))
      timer ← t_sleep
      if (not communication_complete())
        state ← WAITING
        wait_timer ← t_wait_max
      else
        radio_off()
        state ← OFF
  elseif (state = WAITING)
    if (communication_complete() or
        expired(wait_timer))
      state ← OFF
      radio_off()
  elseif (state = OFF)
    if (expired(timer))
      radio_on()
      state ← ON
      timer ← t_awake
```

Example: Protothread implementation

```
radio_wake_protothread:
  PT_BEGIN
  while (true)
    radio_on()
    timer ← t_awake
    PT_WAIT_UNTIL(expired(timer))
    timer ← t_sleep
    if (not communication_complete())
      wait_timer ← t_wait_max
      PT_WAIT_UNTIL(communication_complete() or
                    expired(wait_timer))
    radio_off()
    PT_WAIT_UNTIL(expired(timer))
  PT_END
```

Qualitative comparison

Property	Multi-threading	Event-driven	Protothreads
Memory requirement	Higher	Lower	Lower
Control structures	Yes	No	Yes
Debug stack retained	Yes	No	Yes
Implicit locking	No	Yes	Yes
Preemption	Yes	No	No
Automatic variables	Yes	No	No

Software platform

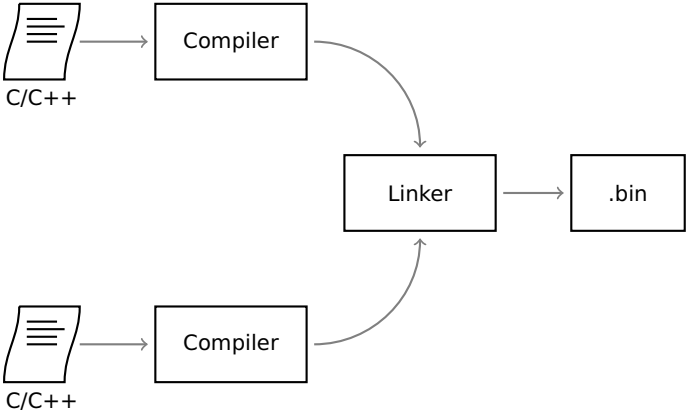
CONTIKI

NEXT GENERATION

Contiki-NG file structure

Folder	Description
os	Operating system code
arch	Hardware dependent code
examples	Example programs
tools	Tools for flashing, simulations, ...
tests	Test codes

The build system



Make and Makefiles

Make

- Build tool (i.e., builds executable programs from sources)
- Sources, tools and options can be configured

Makefile

- Used to configure make

Compiling for specific hardware

Build tool

Specific hardware

```
> make TARGET=zoul BOARD=firefly-reva hello-world
```

Target platform

Makefile target

Starting from scratch

```
> make clean
```

Removes all previously built files from the directory

Uploading the program

```
> make ...  hello-world.upload
```

Uploads the executable file
after building is complete

Reading serial output

```
> make TARGET=zoul BOARD=firefly-reva login
```

Opens a serial connection
to the node