Univerza v Ljubljani

MACHINE VISION GROUP

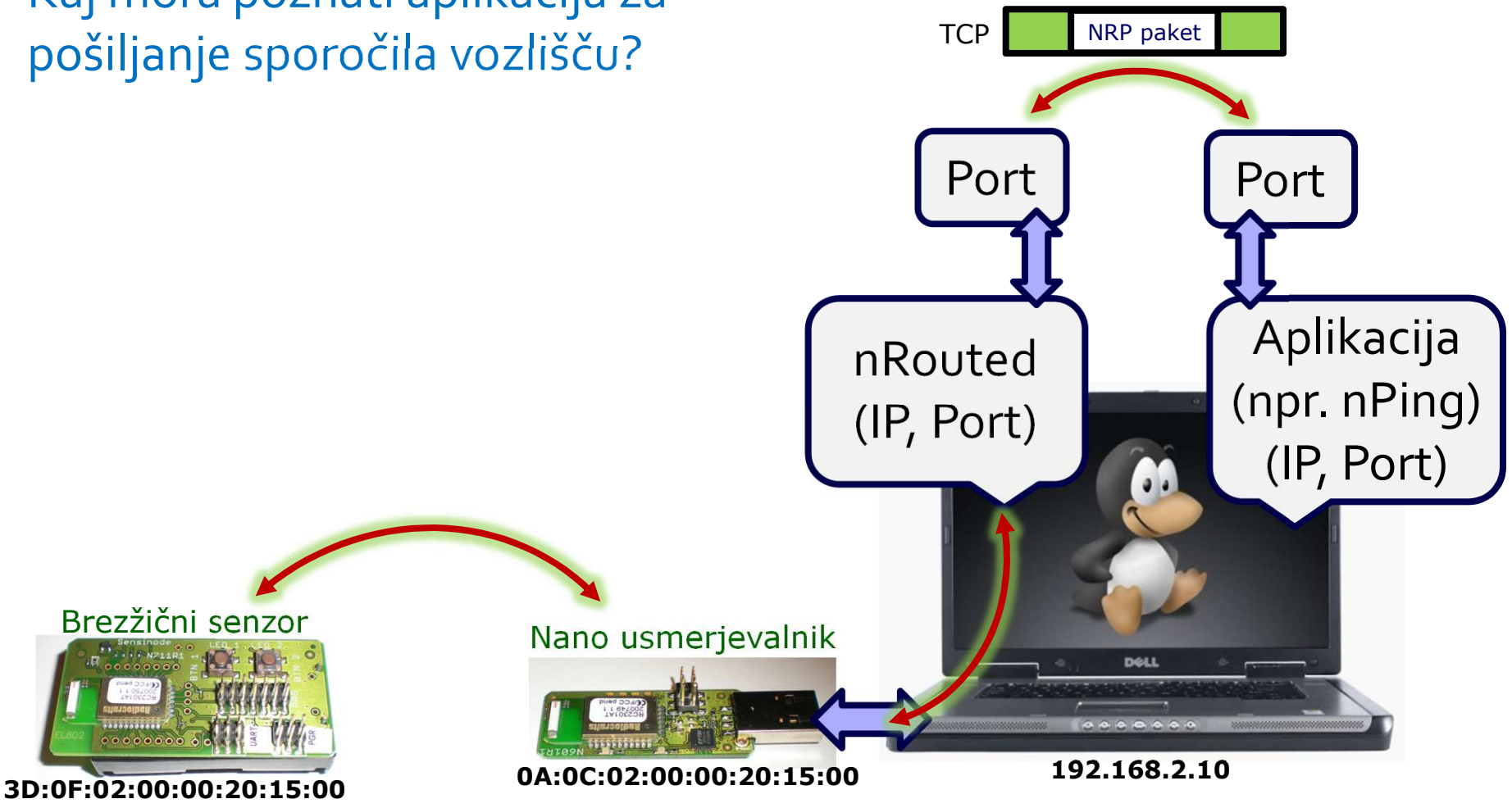
Sistemi Daljinskega Vodenja
Vaja 2

Matej Kristan
<matej.kristan@fe.uni-lj.si>

Laboratorij za Strojni Vid
Fakulteta za elektrotehniko, Univerza v Ljubljani
matej.kristan@fe.uni-lj.si

Česa smo se naučili v prvi vaji: Delovanje

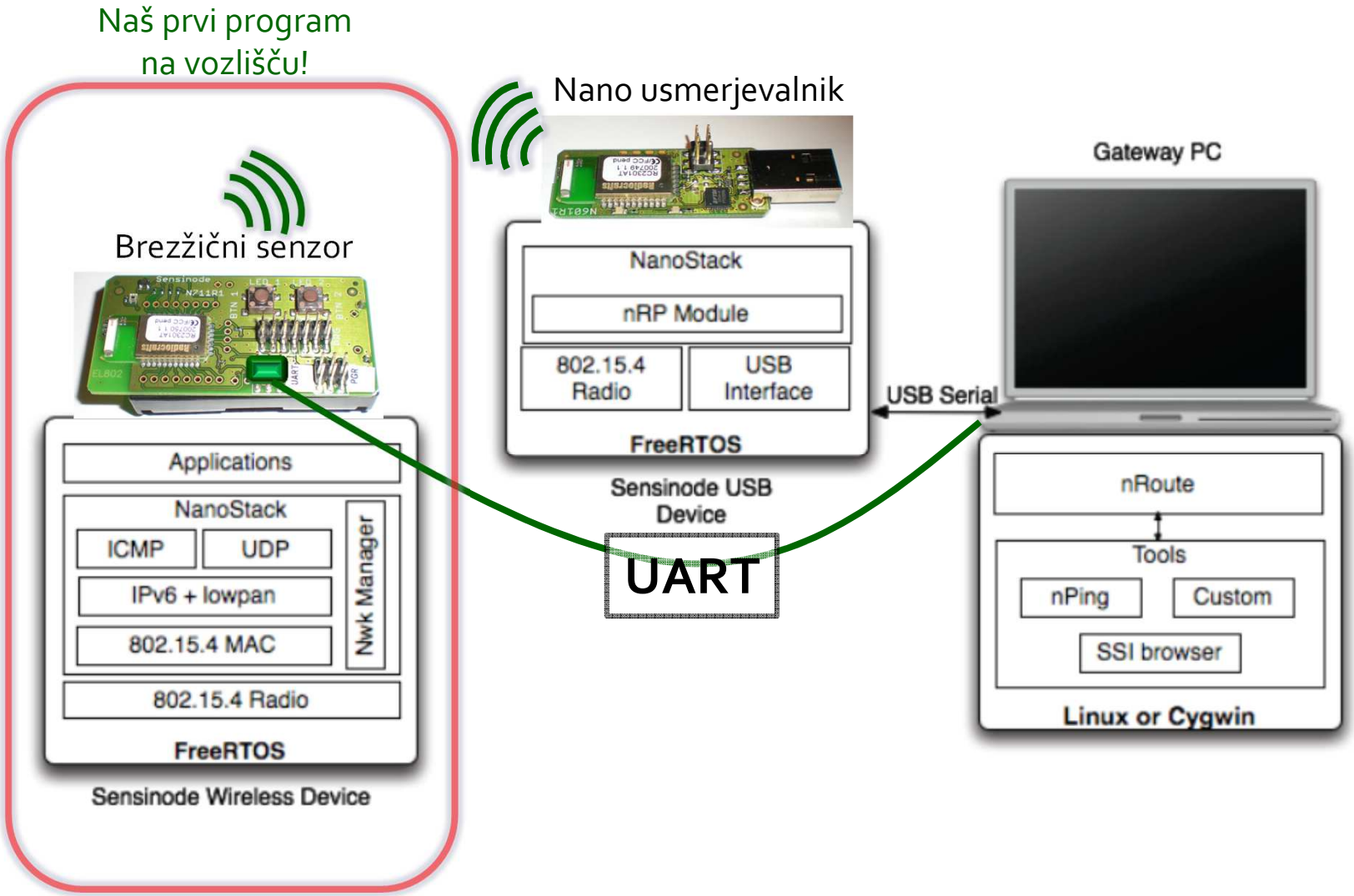
Kaj mora poznati aplikacija za pošiljanje sporočila vozlišču?



Vsebina vaj: prvo programiranje

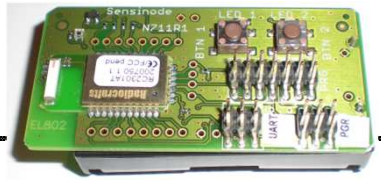
- NanoStack.
- Struktura aplikacij.
- Prvi program.

Hierarhija programske opreme:



Slike povzete po: NanoStack Manual, Sensionode, 2008

Tipična struktura programa



1. vključitve za predprevajalnik

```
#include <stdlib.h>
#include <string.h>
#include <sys/inttypes.h>
```

2. deklaracije

```
// global variables
portTickType    scan_start ;
// my functions
static void vAppTask( int8_t *pvParameters );
```

3. glavna funkcija

```
/* Main task, initialize hardware and sta
int main( void )
{
        /* Initialize the Nano hardware *,
        LED_INIT(); // initialize leds
```

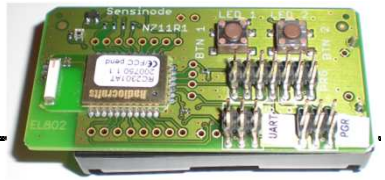
4. definicije naših funkcij

```
static void vAppTask( int8_t *pvParameters )
{
        pvParameters ; // just so that the c
        /* Start the debug UART at 115k */
```

3

```
int main( void )
{
        /* Initialize the Nano hardware */
    ● bus_init() ; // initialize hardware
    ● debug_init(115200); // initialize debug stuff
    ● stack_init(); // initialize stack variables
        // Setup the application task and start the scheduler
    ● xTaskCreate( vAppTask, "App", configMAXIMUM_STACK_SIZE, NULL,
                   (tskIDLE_PRIORITY + 2 ), ( xTaskHandle * )NULL ) ;
    ● vTaskStartScheduler();
        return 0;
}
```

Tipična struktura programa



1. vključitve za predprevajalnik

```
#include <stdlib.h>
#include <string.h>
#include <sys/inttypes.h>
```

2. deklaracije

```
// global variables
portTickType    scan_start ;
// my functions
static void vAppTask( int8_t *pvParameters );
```

3. glavna funkcija

```
/* Main task, initialize hardware and sta
int main( void )
{
        /* Initialize the Nano hardware *
        LED_INIT(); // initialize leds
```

4. definicije naših funkcij

```
static void vAppTask( int8_t *pvParameters )
{
        pvParameters ; // just so that the c
        /* Start the debug UART at 115k */
```

1. Definicije in dodatne inicializacije
2. Glavna zanka, kamor vstavimo program.

4

```
static void vAppTask( int8_t *pvParameters )
{
    pvParameters ; // just so that the compiler doesn't complain.

    // enter infinite loop
    for (;;) {
       /* here comes the main code */


      /* end of the main code */
     /* Sleep for 500 ms */
    vTaskDelay( 500 / portTICK_RATE_MS );
    }

}
```

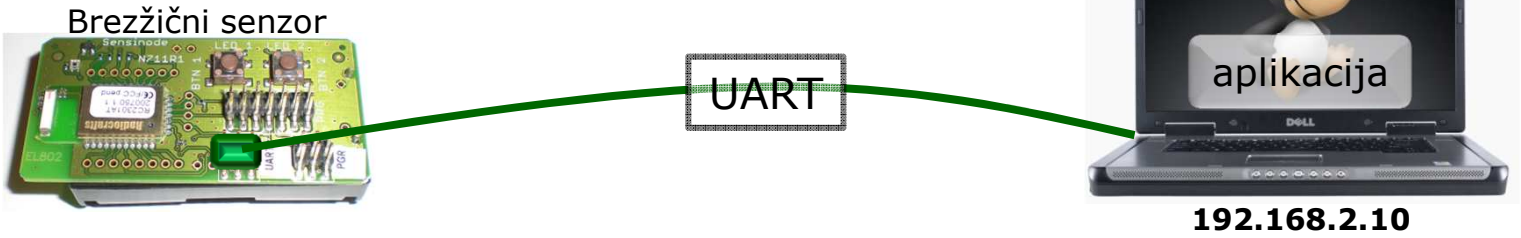
1

2

Nekaj o delovanju in APIju

Obstoječi API za delo z ledicami:

```
LED1_ON(void), LED1_OFF(void), LED1_TOGGLE(void)
LED2_ON(void), LED2_OFF(void), LED2_TOGGLE(void)
```



Brezžični senzorji morajo biti večino časa v latentni fazi (spati):

```
void vTaskDelay( portTickType xTicksToDelay )
```

Merjenje časa v enotah notranje ure:

```
portTickType xTaskGetTickCount(void)      portTICK_RATE_MS
```

Blokiranje in uart:

```
int16_t debug_read_blocking( uint32_t time)
int8_t debug( str )
uint8_t * debug_int(int y)
```

Naloga za danes

- **Izhajamo** iz: **Obtoječega programa** (okostja) brez inicializacije in funkcionalnosti.

- **Vstavili inicializacijo** in demonstrirali:
  1. enostavno komunikacijo preko UART (cuteCom terminal),
  2. zakasnitve, spreminjanje komunikacijskega kanala,
  3. prižiganje ter ugašanje ledic.

- **Naloga je sestavljena iz dveh delov**,
  ko končate en del, me pokličite, da pregledam.

- Izdelek bo povzetek dela v poročilu, v katerem opišete nalogo, API, potek naloge, prav tako boste dodali še rezultate (npr., slike cuteComa).

Konec.

- Hvala.