Let’s Talk!

Jeroen de Best

April 27, 2007

Embedded Motion Control (4K450)

Workshop Sense And Act

Previous workshop

- Read out sensor values

  Rectification on encoder example:

  `sleep(0.1);` should be `msleep(100);`

- Actuate motors
- Few ways to print information on screen

Implemented it?
Are there any problems / questions so far?

Let’s Talk!

Communication is essential for the space mission!

Let’s Talk!

Communication is essential for the space mission!

For example the famous words taken from the Apollo 13 in 1970:

"Houston, we have a problem."

We should avoid this kind of messages...

Let’s Talk!

Why communicate?

- Synchronization[1] between RCX bricks
- ’Interrupt’ tasks[1] that are carried on other RCX brick
- Get coordinates from USB Tower (’Earth’)
- ...

Tool to communicate

- Infrared ports on RCX bricks and USB Tower
- LegOS Network Protocol (LNP)

See workshop "Tasks and Synchronization"

Let’s Talk!

Communication with LNP is prepared for Mars mission.

Requirement: `com.c` (Posted on Studyweb & on EMC CD)

What’s in it?

- Function `read_from_ir`
- Function `com_send`

Natural question arises:

How do I use these functions?

Let’s Talk!

What does `read_from_ir`?

In (short) pseudo code:

```
while(1) {
    /* Two questions here: */
    /* - is there a package? */
    /* - is it destined for me? */
    wait_event(havepack, 0);
    /* If so, then retrieve data */
    ...
    /* Adjust corresponding global variable */
    ...
}
```

Function must be started as a thread / task [1]!

[1] See workshop "Tasks and Synchronization"

Let’s Talk!

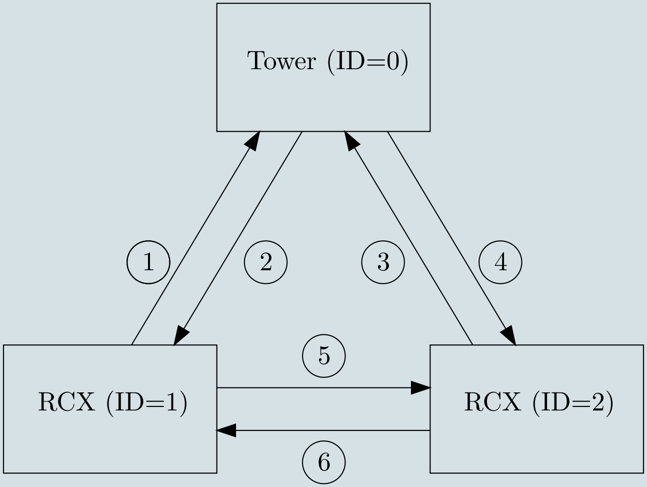
To include the communication in your programs, your main files should look like:

```
/* define ID of this brick */
#define ID 1
/* include unistd.h (see page 167 Extreme Mindstorms)*/
#include <unistd.h>
/* include com.c */
#include "com.c"
/* declare thread id for read_from_ir_thread */
tid_t read_from_ir_thread;

/* begin main */
int main(int argc char *argv[]) {
    /* initialize communication port */
    lnp_integrity_set_handler(port_handler);
    /* set ir range to "far" */
    lnp_logical_range(1);
    /* initialize semaphore for communication */
    sem_init(&sem_com, 0, 1);
    /* start read_from_ir thread */
    read_from_ir_thread = execi(&read_from_ir,0,0,PRIO_NORMAL,DEFAULT_STACK_SIZE);
/* end main */
}
```

Let's Talk!

Possible configuration of communication



ID of USB Tower ID is 0 (fixed).
ID of RCX bricks can be chosen.

Let’s Talk!

Use `com_send` to send a message (integer) to the other RCX brick or USB Tower (’Earth’).

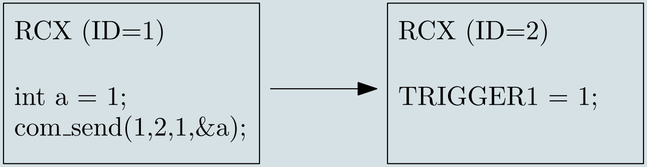
Usage:

```
int com_send(int IDs, int IDr, int message_type, int *message);
```

Example (see communication line 5 in previous scheme)

```
int a = 1;
com_send(1,2,1,&a);
```

At the other end of the communication line (here, at RCX brick with ID 2) this is translated in `TRIGGER1 = 1;`(done by the `read_from_ir` thread running at RCX brick with ID 2)



Let’s Talk!

Global variables defined on each RCX brick correspond to message types

1. TRIGGER1 (integer)
2. TRIGGER2 (integer)
3. TRIGGER3 (integer)
4. TRIGGER4 (integer)
5. TRIGGER5 (integer)
6. TEMPERATURE (integer)
7. COORDINATES (array of 6 integers)

Let’s Talk!

Miscellaneous:

- To get coordinates from ’Earth’ just send any message to the USB Tower.
  Example:

  ```
  com_send(1,0,1,&a);
  ```

- To adjust the global variables yourself use semaphores!
  Example:

  ```
  sem_wait(&sem_com);
  TRIGGER1 = 1;
  sem_post(&sem_com);
  ```

- There is no guarantee that your message arrives correctly...

- 2 Example files posted on Studyweb:
  WS_11042006_RCX1.c & WS_11042006_RCX2.c