Documentación Básica de Comandos de la PC remota a la PC104 y de la PC104 a los PICs

***Revision*:** según código que consta en la carpeta **1004_codigo_completo** que incluye la primer prueba de joystick que tenemos del Exa incluyendo todos los sensores (telemetros, linefollowing, bumper, sonar) + los motores.

Fecha: 9/04/10

Nota: La arquitectura de la solución hasta el momento es la descrita en el paper del JACSM. Tiene 3 threads en la PC104 y 3 en la PC remota: udp_send, udp_receive y main_application. La comunicación entre esos threads es a través de los buffers. Para mayor referencia, leer el paper.

1) Comandos en el joystick

M: motores
ML val -> poner el valor al motor left
MR val -> poner el valor al motor right
MS val_right val_left -> poner el val_left al motor_left, y val_right al motor_right

La velocidad a los motores va entre -30 y 30 (el signo indica dirección).

P: prender sensor
PT1 -> Prender los tres telemetros conectados al AN0, AN1 y AN2 (el AN2 en este momento no tiene telemetro conectado)
PT2 -> Prender los tres telemetros conectados al AN3, AN4 y AN5 (el AN5 en este momento no tiene telemetro conectado)
PT3 -> Prender los dos telemetros conectados al AN6 y AN7.

PS -> Prender Sonar
PB -> Prender Bumpers
PL -> Prender Linefollowing.

A: apagar sensor
AT1 -> Apagar los tres telemetros conectados al AN0, AN1 y AN2 (el AN2 en este momento no tiene telemetro conectado)
AT2 -> Apagar los tres telemetros conectados al AN3, AN4 y AN5 (el AN5 en este momento no tiene telemetro conectado)
AT3 -> Apagar los dos telemetros conectados al AN6 y AN7.

AS -> Apagar Sonar.
AB -> Apagar Bumpers.
AL -> Apagar Linefollowing.

Respuesta:
El thread udp_receive esta recibiendo constantemente e imprime por pantalla el vector recibido de 18 bytes con toda la telemetria del robot, con el siguiente formato:

V[0] = 255 (semaforo de la comunicación entre threads, no es un dato importante)
V[1] = motor_right_valido (240 si es valido el valor que sigue).
V[2] = encoder_motor_right
V[3] = motor_left_valido (240 si es valido el valor que sigue).

V[4] = encoder_motor_left

V[5] = Valor Telemetro conectado en AN0
V[6] = Valor Telemetro conectado en AN1
V[7] = Valor Telemetro conectado en AN2
V[8] = Valor Telemetro conectado en AN3
V[9] = Valor Telemetro conectado en AN4
V[10] = Valor Telemetro conectado en AN5
V[11] = Valor Telemetro conectado en AN6
V[12] = Valor Telemetro conectado en AN7

V[13] = Valor Sonar (byte menos significativo) //sigue la convencion little endian cuac!
V[14] = Valor Sonar (byte mas significativo)

V[15] = Linefollowing conectado al RB2
(contador de hace cuánto que no ve la linea: 1 -> la estoy viendo en este momento, 255 = hace infinito que no la veo).
V[16] = Linefollowing conectado al RB3
(contador de hace cuánto que no ve la linea: 1 -> la estoy viendo en este momento, 255 = hace infinito que no la veo).

V[17] = Bumper conectado al RB1. (255 -> esta apretado, 0 -> esta libre)
V[18] = Bumper conectado al RB4. (255 -> esta apretado, 0 -> esta libre)

2) Comandos de la PC remota a la PC104

En esta sección presentamos el formato de los paquetes enviados de la PC remota a la PC104. Todos los paquetes tienen 4 bytes de longitud, y en esta etapa del código son muy similares a los paquetes que se reenvian tal cual a los PICs. O sea, esto habria que cambiarlo, se cambia modificando únicamente el archivo main_application.c de la PC104 para agregarle toda la inteligencia que uno quiera y traducir paquetes de alto nivel que llegan de la PC remota a los paquetes de bajo nivel que se envian de la PC104 a los PICs.

Comandos para los motores:

Valor para Motor Right
cmd[0] = 0x00;
cmd[1] = (char) val_mot_derecho;
(para valores negativos de valor, es 255 menos el valor, por ej -10 es 245).

Valor para Motor Left
cmd[0] = 0x01;
cmd[1] = (char) val_mot_left;
(para valores negativos de valor, es 255 menos el valor, por ej -10 es 245).

Valor para ambos motores
cmd[0] = 0x03;
cmd[1] = (char) val_mot_derecho;
cmd[2] = (char) val_mot_izquierdo;

Comandos para los sensores

- Todos los comandos para los sensores, empiezan con cmd[0] = 0x02 (indica que el paquete es para los sensores).
- El cmd[1] indica si prender o apagar sensores:
    - cmd[1] = 0x01 (indica que se prenden sensores).
    - cmd[1] = 0x02 (indica que se apagan sensores).
- Los siguientes bytes son para especificar qué sensor prender o apagar:
    - cmd[2] = 0x00 -> telemetros conectados al AN0, AN1 y AN2
    - cmd[2] = 0x01 -> telemetros conectados al AN3, AN4 y AN5
    - cmd[2] = 0x02 -> telemetros conectados al AN6, AN7
    - cmd[2] = 0x03 -> sonar
    - cmd[2] = 0x04 -> linefollowing
    - cmd[2] = 0x05 -> bumper

Respuesta de la PC104

10 veces por segundo, la PC104 responde con toda la telemetria disponible, siguiendo el vector ya especificado de 18 bytes, pero que aquí repito:

V[0] = motor_right_valido (240 si es valido el valor que sigue).
V[1] = encoder_motor_right
V[2] = motor_left_valido (240 si es valido el valor que sigue).
V[3] = encoder_motor_left

V[4] = Valor Telemetro conectado en AN0
V[5] = Valor Telemetro conectado en AN1
V[6] = Valor Telemetro conectado en AN2
V[7] = Valor Telemetro conectado en AN3
V[8] = Valor Telemetro conectado en AN4
V[9] = Valor Telemetro conectado en AN5
V[10] = Valor Telemetro conectado en AN6
V[11] = Valor Telemetro conectado en AN7

V[12] = Valor Sonar (byte menos significativo) //sigue la convencion little endian cuac!
V[13] = Valor Sonar (byte mas significativo)

V[14] = Linefollowing conectado al RB2
(contador de hace cuánto que no ve la linea: 1 -> la estoy viendo en este momento, 255 = hace infinito que no la veo).
V[15] = Linefollowing conectado al RB3
(contador de hace cuánto que no ve la linea: 1 -> la estoy viendo en este momento, 255 = hace infinito que no la veo).

V[16] = Bumper conectado al RB1. (255 -> esta apretado, 0 -> esta libre)
V[17] = Bumper conectado al RB1. (255 -> esta apretado, 0 -> esta libre)

Nota: La PC104 tiene un modo debug que solo envia datos 1 vez por segundo para poder ver el printf de los datos. También puede programarse para que envie datos mas seguidos, pero más de 33 veces por segundo no tiene sentido porque no tendrá telemetria nueva que mandar, solo repetirá el envio anterior.

3) Paquetes SPI de la PC104 a los PICs.

Paquetes para setear velocidad a motor right
paquete_out_mr[0] = 0x10;

paquete_out_mr[1] = velocidad (entre -30 y 30)

Paquetes para setear velocidad a motor left
paquete_out_ml[0] = 0x10;
paquete_out_ml[1] = velocidad; (entre -30 y 30)

Paquetes a sensores

- El paquete_out_sn[0] indica si prender o apagar sensores:
  - paquete_out_sn[0] = 0x01 (indica que se prenden sensores).
  - paquete_out_sn[0] = 0x02 (indica que se apagan sensores).
- Los siguientes bytes son para especificar qué sensor prender o apagar:
  - paquete_out_sn[1] = 0x00 -> telemetros conectados al AN0, AN1 y AN2
  - paquete_out_sn[1] = 0x01 -> telemetros conectados al AN3, AN4 y AN5
  - paquete_out_sn[1] = 0x02 -> telemetros conectados al AN6, AN7
  - paquete_out_sn[1] = 0x03 -> sonar
  - paquete_out_sn[1] = 0x04 -> linefollowing
  - paquete_out_sn[1] = 0x05 -> bumper

Respuestas:

Paquetes de respuesta del motor right
paquete_in_mr[0] = valido (240).
paquete_in_mr[1] = valor del encoder.

Paquetes de respuesta del motorleftt
paquete_in_ml[0] = valido (240).
paquete_in_ml[1] = valor del encoder.

Paquetes de sensores
paquete_in_sn[0] -> telemetria de qué sensor:

```
paquete_in_sn[0] = 0x00  //primeros tres telemetros
paquete_in_sn[0] = 0x01  //segundos tres telemetros
paquete_in_sn[0] = 0x02  //terceros dos telemetros
paquete_in_sn[0] = 0x03 //sonar
paquete_in_sn[0] = 0x04  //linefollowing
paquete_in_sn[0] = 0x05  //bumper
```

los otros 3 bytes son los valores de esos sensores.

- ➔ en el caso de los telemetros, un byte por telemetro
- ➔ en el caso del sonar,
  paquete_in_sn[1] = byte menos significativo
  paquete_in_sn[2] = byte mas significativo
- ➔ en el caso del linefollowing
  paquete_in_sn[1] = linefollowing conectado al RB2
  paquete_in_sn[2] = linefollowing conectado al RB3
- ➔ en el caso del bumper
  paquete_in_sn[1] = Bumper conectado al RB1.
  paquete_in_sn[2] = Bumper conectado al RB4.